

TABLE OF CONTENTS

What's New	1
Administrative Tasks	
 Microsoft EULA	10
 PNA User Accounts and Passwords	31
 Computer Properties	34
 Error-check and Disk Defragmenter	38
 Operating System Recovery	40
 Wake PNA at Specified Time	41
 Windows Considerations	43
Quick Start	
 Front Panel Tour	45
 Rear Panel Tour	55
 Powering the PNA ON and OFF	60
 Traces, Channels, and Windows on the PNA	63
 QuickStart Dialog	71
 Basic Measurement Sequence	77
 Frequency Blanking	78
 Internal Second Source	81
 Connectivity Guide	83
 PNA Preferences	84
 VNC	90
 LXI Compliance	92
 Windows 7 File Locations	95
 Dialog Transparency	96
 Using Help	98
 Help About	106
 Searching PNA Help	108
1. Set Up a Measurement	
 Preset the PNA	109
 Measurement Classes	111
 Measurement Parameters	114

Frequency Range	123
Power Level	132
Receiver Leveling	143
Sweep Settings	150
Trigger Setup	162
External Triggering	167
Trigger Model Animation	174
Data Format	175
Scale	182
Preconfigured Measurement Setups	188
Path Configurator	192
Phase Control	196
Customize Your Analyzer Screen	204
Copy Channels	215
DC Control	219
ADC Measurements	222
Undo/Redo	225
2. Optimize a Measurement	
Dynamic Range	228
Number of Data Points	231
Phase Accuracy	233
Phase Coherent Measurements	238
Electrically Long Devices	240
Reflection Accuracy	243
Measurement Stability	246
Noise Reduction Techniques	248
Crosstalk	257
Effects of Accessories	258
Fastest Sweep	259
Multiple State Measurements	261
Fastest Data Transfer	265
Using Macros	266
3. Calibrate a Measurement	

Select a Calibration	271
Calibration Wizard	275
Guided Power Calibration	292
Calibrate All Channels	297
Using Cal Sets	307
Error Correction and Interpolation	316
ECal	
Using ECal	321
ECal User Characterization	331
Perform a 4-Port Cal with a 2-Port ECal Module	347
TRL Calibration	349
CalPod	354
CalPod as ECal	362
Calibration Preferences	372
Power Calibration	377
Fixture Compensation	397
Port Extensions	413
Characterize Adaptor Macro	422
Swap Adapters Method	431
Delta Match Calibration	433
Concepts	
Calibration Overview	438
Measurement Errors	441
Accurate Calibrations	455
Calibration Thru Methods	459
Validity of a Cal	465
Calibration Standards	471
Modify Cal Kits	476
Why Modify a Cal Kit	481
About PNA Cal Kits and Firmware Upgrades	482
Creating Custom Calibration Kits using a New Connector Family	483
How to Create a New Cal Kit from an Existing Cal Kit	485
VNA Cal Kit File Types	486

Connectors Tab	488
Standards Tab	491
SOLT Tab	500
TRL Tab	503
4. Analyze Data	
Locate Data Using Markers	507
Math & Memory Operations	525
8510 Mode	530
Equation Editor	531
Use Limits to Test Devices	546
5. Output Data	
Save and Recall Data	551
Drive Mapping	572
Print	573
Programming	579
COM	
Commands	
Objects	
The Analyzer Object Model	580
Application Object	583
AuxTrigger Object	590
BalancedMeasurement Object	593
BalancedStimulus Object	595
BalancedTopology Object	597
CalFactorSegments Collection	600
CalFactorSegmentsPMAR Collection	601
CalibrateAllChannels Object	603
Calibrator Object	606
CalKit Object	612
CalManager Object	614
CalSet Object	617
CalSets Collection	622
CalStandard Object	624

 Capabilities Object	627
 Channel Object	630
 Channels Collection	638
 ComColors Object	640
 ComTraceColors Object	642
 Converter_Object	644
 ConverterEmbeddedLO Object	650
 CorrectionMethods Object	652
 DCStimulus Object	654
 E5091Testset Collection	656
 E5091Testset Object	658
 ECalModule Object	660
 ECalModules Collection	662
 ECalUserCharacterizer Object	664
 EmbeddedLO Object	667
 EmbeddedLODiagnostic Object	669
 ENRFile Object	671
 Equation Object	673
 ExternalDCDevice Object	675
 ExternalDevice Object	678
 ExternalDevices Collection	681
 ExternalPulseGenerator Object	683
 ExternalSource Object	685
 ExternalTestsets Collection	687
 FIFO Object	689
 Fixturing Object	691
 FOM Collection	696
 FOMRange Object	698
 GainCompression Meas Object	701
 GainCompression Object	703
 GainCompressionCal Object	707
 Gating Object	709
 GlobalPowerLimit Object	711

GroupDelayAperture Object	713
GuidedCalibration Object	715
GuidedCalibrationPowerSensor Object	720
GuidedCalibrationPowerSensor Object#history	720
GuidedCalibrationPowerSensors Collection	722
HWAuxIO Object	724
HWExternalTestSetIO Object	727
HWMaterialHandlerIO Object	729
IIFConfiguration Object	731
IMixer Interface	733
IMSpectrum Object	737
InterfaceControl Object	739
Limit Test Collection	740
LimitSegment Object	742
Marker Object	744
Measurement Object	748
MeasurementClassProperties Object	756
Measurements Collection	758
NAWindow Object	759
NAWindows Collection	761
NoiseCal Object	762
NoiseFigure Object	765
PathConfiguration Object	768
PathConfigurationManager Object	770
PathElement Object	772
PhaseControl Object	774
PhaseReferenceCalibration Object	776
PNOP Object	779
Port Extension Object	781
PowerLossSegment Object	783
PowerLossSegments Collection	785
PowerLossSegmentPMAR Object	786
PowerLossSegmentsPMAR Collection	788

 Active Marker	857
 Active Measurement	858
 Active NAWindow	859
 Active Trace	860
 ActiveBackground Property	861
 ActiveLabels Property	862
 ActiveXAxisRange	863
 ActiveXAxisRange(Conv)	864
 ADCCaptureMode	866
 ALCLevelingMode	867
 AllowArbitrarySegments	869
 Alternate Sweep	870
 AmbientTemperature	871
 AnalysisCWFreq Property	872
 AnalysisEnable Property	873
 AnalysisIsDiscreteFreq Property	874
 AnalysisXAxis Property	875
 Application	876
 Arrange Windows	877
 Attenuator Mode	878
 Attenuator	879
 AutoCWSweepTime Property	881
 AutoDetection Property	882
 AutoIFBandWidth Property	883
 AutoIFBWAdjustment	884
 AutoIFGain Property	885
 AutoOptimizePRF Property	886
 AutoOrient	887
 AutoOrientTuner Property	888
 AutoPortExtConfig	889
 AutoPortExtDCOffset	890
 AutoPortExtLoss	891
 AutoPortExtSearchStart	892

 AutoPortExtSearchStop	893
 AutoPortExtState	894
 AutoPulseTiming Property	895
 AutoSelectPulseGen Property	896
 AuxiliaryTriggerCount	898
 AuxTriggerScopelsGlobal	899
 AvailableMeasurementClasses Property	901
 AverageMode	902
 Averaging Count	903
 Averaging Factor	904
 Averaging ON/OFF	905
 AvoidSpurs	906
 Background Property	907
 BackOff Property	908
 BackOffGain Property	909
 BackOffPIn Property	910
 BackOffPout Property	911
 BalancedMode	912
 BalPort1PhaseOffset	913
 BalPort1PowerOffset	914
 BalPort1StartPhase	915
 BalPort1StopPhase	916
 BalPort2PhaseOffset	917
 BalPort2PowerOffset	918
 BalPort2StartPhase	919
 BalPort2StopPhase	920
 Bandwidth Target	921
 Bandwidth Tracking	922
 BB_BalPort1Negative	924
 BB_BalPort1Positive	925
 BB_BalPort2Negative	926
 BB_BalPort2Positive	927
 BalSMeasurement Property	928

CharacterizeMixerOnly	967
CharFileName	968
CharMixerReverse	969
CitiContents	970
CitiFormat	971
CmnModeZConvPortImag	972
CmnModeZConvPortReal	973
CmnModeZConvPortZ0	974
CmnModeZConvState	975
CompatibleCalKits	976
CompositeNormalizationMode	978
CompositeNormalizedCSOPower	980
CompositeNormalizedCTBPower	981
Compression Property	982
CompressionAlgorithm	983
CompressionBackoff	984
CompressionDeltaX	985
CompressionDeltaY	986
CompressionInterpolation	987
CompressionLevel	988
CompressionMax Property	989
CompressionLevel (mkr)	990
CompressionPin	991
CompressionPout	992
CompressionSaturation Property	993
ConnectorType	994
ConnectorType ECal	996
ControlLines	998
Count	1000
Couple Ports	1002
CoupleChannelParams	1003
Coupled	1004
Coupled Markers	1005

CoupledParameters - Gate	1006
CoupledParameters - Transform	1007
CouplePhasePortSettings Property	1009
CoupleTonePower	1010
CpuRevision Property	1011
CSONumDistortionProducts	1012
CSOffset	1013
CTBOffset	1014
CTBXMODNumCarriers	1015
CustomCalConfiguration Property	1016
CustomChannelConfiguration	1017
CustomMeasurementConfiguration Property	1018
CW Frequency	1019
CWFrequency CS Property	1020
Data	1021
DataAndLimits Property	1022
DataCount	1023
DataInCompactForm	1024
DCCorrection Property	1025
DCOffset Property	1026
DCScale Property	1027
DCType Property	1028
DefinedRoles Property	1029
Delay	1030
Delay pulse	1031
Delay_trigger	1032
DelayCalculationMethod Property	1033
DelayIncrement	1034
DeltaFrequency	1036
DeltaFrequencyStart	1037
DeltaFrequencyStop	1038
DeltaMarker	1039
Description	1040

 DescriptiveText	1041
 DeviceInputPort (FCA)	1042
 DeviceInputPort	1043
 DeviceNames Property	1044
 DeviceOutputPort (FCA)	1045
 DeviceLinearPowerLevel	1046
 DeviceOutputPort	1047
 DeviceType Property	1048
 DiffPortMatch_C	1049
 DiffPortMatch_G	1050
 DiffPortMatch_L	1051
 DiffPortMatch_R	1052
 DiffPortMatchMode	1053
 DiffPortMatchState	1054
 DiffPortMatchUserFilename	1055
 DiffZConvPortImag	1056
 DiffZConvPortReal	1057
 DiffZConvPortZ0	1058
 DiffZConvState	1059
 Display Format	1060
 DisplayAutomationErrors	1062
 DisplayGlobalPassFail	1063
 DisplayRange	1064
 Distance	1065
 DistanceMarkerMode	1066
 DistanceMarkerUnit	1068
 Divisor	1069
 Do1PortEcal	1070
 Do2PortEcal	1071
 Domain	1072
 Driver Property	1073
 DspFpgaRevision Property	1074
 DspRevision Property	1075

 DUTTopology	1076
 Dwell Time	1077
 DwellPerPoint Property	1078
 ECALCharacterization (smc)	1079
 ECALCharacterization (vmc)	1081
 ECALCharacterizationEx	1083
 ECallD	1085
 ECALIsolation	1086
 ECALModuleNumberList	1087
 EcalOrientation	1088
 EcalOrientation1Port	1090
 EcalOrientation2Port	1091
 ECALPortMapEx	1093
 ElecDelay Medium	1095
 ElecDistanceDelay	1096
 ElecDistanceDelayUnit	1097
 Electrical Delay	1098
 Element	1099
 Elements	1100
 Embed4PortA	1101
 Embed4PortB	1102
 Embed4PortC	1103
 Embed4PortD	1104
 Embed4PortList	1105
 Embed4PortNetworkFilename	1107
 Embed4PortNetworkMode	1108
 Embed4PortState	1109
 Embed4PortTopology	1110
 Enable	1111
 EnableAllOutput Property	1112
 Enabled	1113
 EnableLOPowerCal	1114
 EnablePhase Property	1116

 EnablePowerCompensation Property	1117
 EnableSnPDDataExtrapolation Property	1118
 EnableSourceUnleveledEvents	1119
 EndOfSweepOperation	1120
 ENRFile	1121
 ENRID	1122
 ENRSN	1123
 EqualTonePower Property	1124
 Error Correction	1125
 ErrorCorrection(Channel)	1126
 ErrorCorrectionIndicator Property	1127
 ExtendedProperties Property	1128
 External ALC	1129
 ExternalDeviceDeActivatePolicy Property	1130
 ExternalTriggerConnectionBehavior	1131
 ExternalTriggerDelay	1134
 F1Frequency	1135
 F2Frequency	1136
 FailedTraces Property	1137
 FastCWPointCount	1138
 FastMode	1139
 Filter BW	1140
 Filter CF	1141
 Filter Loss	1142
 Filter Q	1143
 FilterErrors	1144
 FilterMode	1146
 FirmwareMajorRevision	1147
 FirmwareMinorRevision	1148
 FirmwareSeries	1149
 FixedDelay Property	1150
 FixedPhase Property	1151
 FixedRatioedPower Property	1152

 FixturingState	1153
 FootSwitch	1154
 Footswitch Mode	1155
 ForceDeEmbedENRAdapter Property	1157
 ForceDeEmbedSensorAdapter Property	1158
 Format (Marker)	1159
 Format	1060
 FormatUnit	1161
 Frequency GD Property	1162
 Frequency	1163
 FrequencyCenter	1164
 FrequencyCenterCenter	1165
 FrequencyCenterspan	1166
 FrequencyCenterStart	1167
 FrequencyCenterStop	1168
 Frequency Offset Divisor	1169
 Frequency Offset Frequency	1170
 Frequency Offset Multiplier	1171
 FrequencyOffsetRangeForCalComputations Property	1172
 Frequency Offset Override To CW	1173
 Frequency Offset State	1175
 FrequencyType Property	1176
 Gain Property	1177
 GainLinear Property	1178
 GainMax Property	1179
 GainSaturation Property	1180
 GeneratedCalsets Property	1181
 FrequencySpan	1182
 Gate Shape	1183
 Gate Type	1184
 GPIBAddress	1185
 GPIB Mode	1186
 GPIBPortCount	1187

 Grid Property	1188
 GridLineType Property	1189
 HandshakeEnable	1190
 HasItem Property	1191
 HighAmplitude Property	1192
 HighestOrderProduct	1193
 ID	1194
 IDString	1195
 IF Bandwidth Option	1196
 IF Bandwidth	1197
 IFBW Property	1198
 IFBWList Property	1199
 IFDenominator_Property	1200
 IFNumerator_Property	1201
 IFFrequency	1202
 IFFrequencyMode	1203
 IFSideband	1204
 IFSideband (conv)	1206
 IFStartFrequency	1208
 IFStopFrequency	1209
 ImpedanceStates	1210
 Impulse Width	1211
 IMToneIFBandwidth	1212
 InactiveLabels Property	1213
 Include2ndOrderProduct	1214
 IncludeReverseSweep Property	1215
 IndexState	1216
 Input A	1217
 Input B	1218
 Input C	1219
 InputDenominator	1220
 InputFixedFrequency	1221
 InputsGreaterThanLO	1222

InputNumerator	1224
InputPower	1225
InputRangeMode	1226
InputRangeMode cv Property	1227
InputStartFrequency	1228
InputStartPower	1229
InputStopFrequency	1230
InputStopPower	1231
InternalTestsetPortCount	1232
Interpolate Correction	1233
Interpolated	1234
InterpolateNormalization	1235
Interrupt	1236
Invert Property	1237
IOConfiguration Property	1238
IOEnable Property	1239
IsContinuous	1241
IsDevicePresent Property	1242
IsECALModuleFoundEx	1243
IsFrequencyOffsetPresent	1244
IsHold	1245
IsMarkerOn	1246
IsolationAveragingIncrement	1247
IsOn	1248
IsReceiverStepAttenuatorPresent	1249
IsReferenceBypassSwitchPresent	1250
IsSParameter	1251
Item Property	1252
Items Property	1255
IterationNumber	1256
IterationsTolerance	1257
Kaiser Beta	1258
L0	1259

 L1	1260
 L2	1261
 L3	1262
 Label	1263
 Label Testset	1264
 LANConfiguration	1265
 LastCalPassedTolerance	1266
 LastLevelingAsSPC	1267
 LastModified	1268
 LevelingIFBW	1269
 Limit Property	1270
 LimitFrequency Property	1271
 LimitTestFailed	1272
 Limit Line Begin Stimulus	931
 Limit Line End Stimulus	1273
 Limit Line Begin Response	930
 Limit Line End Response	1274
 Limit Type	1275
 Line Display	1276
 ListData Property	1277
 LoadCharFromFile	1279
 LoadImpedance Property	1280
 LoadPort	1281
 LocalLockoutState	1282
 Locator	1283
 Lock Property	1285
 LODeltaFound	1286
 LODenominator	1287
 LOFixedFrequency	1288
 LOFrequencyDelta	1289
 OBS_LogMagnitudeOffset	1290
 LOName	1291
 LONumerator	1293

LOPower	1294
LORangeMode	1295
LORangeMode cv Property	1296
Loss	1297
Loss (sourceCal)	1298
LOStage	1299
LOStartFrequency	1300
LOStartPower	1301
LOStopFrequency	1302
LOStopPower	1303
LowAmplitude Property	1304
LXIDeviceIDState	1305
MagnitudeOffset	1307
MagnitudeSlopeOffset	1308
MainToneIFBandwidth	1309
Marker Annotation	1310
Marker Bucket Number	936
Marker Format (all)	1311
Marker Format (indiv)	1159
Marker Interpolate(all)	1313
Marker Interpolate (indiv)	1234
Marker Number	1314
Marker Position	1315
Marker Readout	1316
MarkerReadoutResponsePlaces Property	1317
Marker ReadoutSize	1318
MarkerReadoutsPerTrace Property	1319
MarkerReadoutStimulusPlaces Property	1320
MarkerReadoutXPosition Property	1321
MarkerReadoutYPosition Property	1322
Markers Property	1323
Marker State	1324
MarkerSymbol Property	1325

MasterFrequency Property	1326
MasterMode Property	1327
MasterPeriod Property	1328
MasterWidth Property	1329
Marker Type	1330
Marker X-axis Value	1331
Marker Y-axis Value	1332
MatchCorrectPower	1334
Maximum Frequency	1335
MaximumFrequency (capabilities)	1336
MaximumFrequency (sourceCal)	1337
MaximumIFFrequency	1338
MaximumIterationsPerPoint	1339
MaximumNumberOfChannels	1340
MaximumNumberOfTracesPerWindow	1341
MaximumNumberOfWindows	1342
MaximumNumberOfPoints	1343
MaximumPowerCompression Property	1344
MaximumReceiverStepAttenuator	1345
MaximumSourceALCPower	1346
MaximumSourceStepAttenuator	1347
MaxPreciseTuningIterations	1348
MaxProduct	1349
Mean	1350
Measurement Property	1351
MeasurementClassProperties Property	1352
MeasurementClass	1353
Medium	1354
Memory Property	1355
MemoryMarkers Property	1356
Message Text Method	1357
Minimum Frequency	1358
MinimumFrequency (capabilities)	1359

MinimumFrequency (sourceCal)	1360
MinimumIFFrequency Property	1361
MinimumNumberOfPoints	1362
MinimumReceiverStepAttenuator	1363
MinimumSourceALCPower	1364
MixerCharacterizationFile Property	1365
Mode	1366
Mode-iTMSA	1367
Move Property	1368
Multiplier	1369
Name (Calset)	1370
Name (CalKit object)	1371
Name config	1372
Name element	1373
Name ExtDev Property	1374
Name FOMRange	1375
Name (meas)	1376
Name (PowerSensor)	1377
Name (trace)	1378
NetworkFilename	1379
NetworkMode	1380
NetworkPortMapA Property	1381
NetworkPortMapB Property	1382
NetworkPortMapC Property	1383
NetworkPortMapD Property	1384
NoiseAverageFactor	1385
NoiseAverageState	1386
NoiseBandwidth	1387
NoiseReceiver Property	1388
NoiseReceiverSweepTime Property	1389
NoiseGain	1391
NoiseSourceCalKitType	1392
NoiseSourceCold	1393

NoiseSourceConnectorType	1394
NoiseSourceState	1395
NoiseTuner	1396
NoiseTunerIn	1397
NoiseTunerOut	1398
NominalIncidentPowerState	1399
NormalizePoint Property	1400
NormalizePoint SMC Property	1401
Number (meas)	1402
Number	1403
NumberOfFrequencyPoints	1404
Number of Points	1405
Number of Points (Meas)	1407
NumberOfPorts	1408
NumberOfPorts(Testset)	1409
NumberOfPowerPoints	1410
NumberOfSweeps	1411
Offset	1412
OffsetReceiverAttenuator	1413
OffsetSourceAttenuator	1414
OmitIsolation	1415
OneReadoutPerTrace	1416
Options	1417
OrientECALModule	1418
OutputChannel Property	1420
OutputFixedFrequency	1421
OutputPort	1422
OutputPorts calset	1424
OutputPorts	1425
OutputRangeMode	1426
OutputRangeMode cv Property	1427
OutputSideband (conv)	1428
OutputSideband	1429

OutputStartFrequency	1431
OutputStopFrequency	1432
Parameter	1433
Parameter_elo	1434
Parent	1435
PassFailLogic	1437
PassFailMode	1438
PassFailPolicy	1439
PassFailScope	1440
PassFailStatus	1441
Path Property	1443
PathCalMethod	1444
PathConfigurationElement Property	1446
PathThruMethod	1447
Peak Excursion	1449
Peak Threshold	1450
PeakTo Peak	1451
Percent Property	1452
PerformPowerCalibration	1453
Period	1454
Phase Offset	1455
PhaseAsFixture	1456
PhaseControlMode Property	1457
PhaseCorrectionData Property	1459
PhaseCorrectionEnabled Property	1460
PhaseIterationNumber Property	1461
PhaseParameter Property	1462
PhaseParameterModes Property	1463
PhaseReference Property	1464
PhaseReferencePort Property	1465
PhaseSwpAsFixture	1466
PhaseSwpState	1467
PhaseTolerance Property	1468

Pin Property	1469
PinOffset Property	1470
PMaxBackOff Property	1471
PMaxIn Property	1472
PMaxOut Property	1473
PointAveragingState	1474
PointDwell Property	1475
Points Property	1476
PointSweepState	1477
Port 1	1478
Port 2	1479
Port 3	1480
Port1NoiseTunerSwitchPresetsToExternal	1481
Port2PdeembedCktModel	1482
Port2PdeembedState	1483
PortArbzImag	1484
PortArbzReal	1485
PortArbzState	1486
PortArbzZ0	1487
PortCatalog	1488
PortCLogic	1489
PortCMode	1490
PortCoupleToSystemMedia	1491
PortCoupleToSystemVelocity	1492
PortDelay	1493
PortDescription	1494
PortDistance	1495
PortDistanceUnit	1496
PortExtState	1497
PortExtUse1	1498
PortExtUse2	1499
PortFreq1	1500
PortFreq2	1501

 Port Label	1502
 PortLogic	1503
 PortLoss1	1504
 PortLoss2	1505
 PortLossDC	1506
 PortMatching_C	1507
 PortMatching_G	1508
 PortMatching_L	1509
 PortMatching_R	1510
 PortMatchingCktModel	1511
 PortMatchingState	1512
 PortMedium	1513
 PortMode	1514
 PortsNeedingDeltaMatch	1515
 PortVelocityFactor	1516
 PortWGCutoffFreq	1517
 POut Property	1518
 Power Slope	1519
 Power Acquisition Device	1520
 PowerAsFixture	1521
 PowerCalibrationLevel Property	1522
 PowerCalibrationPowerLevel Property	1523
 PowerLevel	1524
 PowerLevel (CalAll) Property	1525
 PowerMax	1526
 PowerMaxIn Property	1527
 PowerMaxOut Property	1528
 Power Meter Channel	1529
 Power Meter GPIBAddress	1530
 PowerMin	1531
 PowerOffset	1532
 PowerOffset (CalAll) Property	1533
 PowerOnDuringRetraceMode	1534

 PowerSensorCalKitType Property	1535
 PowerSensorCalKitType	1536
 PowerSensorConnectorType	1537
 PowerSensorConnectorType Property	1538
 PowerSlopeState Property	1539
 PowerStep	1540
 PowerSweepRetracePowerMode	1541
 PreciseTuningTolerance	1542
 PreferInternalTriggerOnChannelSingle	1543
 PreferInternalTriggerOnUnguidedCal	1545
 PresetPowerState Property	1547
 PropertyNames Property	1548
 PropertyValue Property	1549
 PropertyValues Property	1550
 PulseGeneratorID Property	1551
 PulseGeneratorNames Property	1552
 PulseMeasMode Property	1553
 RangeCount	1555
 rangeNumber	1556
 RatioedPowerCorrectionData Property	1557
 RatioedPowerCorrectionEnabled Property	1558
 RcvCharMethod Property	1559
 R1 Input Path	1560
 Readings Per Point	1561
 ReadingsTolerance	1562
 ReadyForTriggerPolarity	1563
 ReadyForTriggerState	1564
 RecallSoftkeysMostRecent Property	1565
 Receiver Attenuator	1566
 ReceiverAttenuator Property	1567
 ReceiverCount	1568
 ReceiverRatio Property	1569
 ReceiverStepAttenuatorStepSize	1571

 ReceiverTemperature Property	1572
 Receive Port	1573
 RedTraceOnFail Property	1574
 ReduceIFBandwidth	1575
 ReferenceCalFactor	1576
 Reference Marker State	1577
 ReferenceReceiver	1578
 Reference Level	1580
 Reference Position	1581
 RemoteCalStoragePreference	1582
 ReportReceiverOverload	1584
 ResBWList Property	1585
 ResolutionBW	1586
 Reverse2PortAdapter Property	1587
 ReverseLinearPowerLevel	1588
 RFOffOnReceiverOverload	1589
 RoleDevice Property	1590
 SafeMode	1591
 SafeSweepCoarsePowerAdjustment	1592
 SafeSweepEnable	1593
 SafeSweepFinePowerAdjustment	1594
 SafeSweepFineThreshold	1595
 SafeSweepMaximumLimit Property	1596
 SaturationLevel Property	1597
 SB_BalPortNegative	1598
 SB_BalPortPositive	1599
 SB_SEPort	1600
 SBalMeasurement	1601
 ScaleCouplingMethod Property	1602
 ScaleCouplingState Property	1603
 Scope	1604
 Search Function	1605
 SearchFailures Property	1607

 SearchSummary Property	1608
 SecurityLevel	1609
 Segment Number	1610
 SegmentCount Property	1611
 SegmentFixedFrequency Property	1612
 SegmentFixedPower Property	1614
 SegmentIFBandwidth Property	1616
 SegmentIsInputGreaterThanLO Property	1617
 SegmentMixingMode Property	1619
 SegmentPoints Property	1621
 SegmentRangeMode Property	1622
 SegmentStartFrequency Property	1624
 SegmentState Property	1626
 SegmentStopFrequency Property	1627
 SelectPort	1629
 SensorIndex Property	1630
 SeparatePowerCal Property	1631
 Show Statistics	1632
 ShowProperties	1633
 SICL	1634
 SICLAddress	1635
 Simultaneous2PortAcquisition	1636
 SmartSweepMaximumIterations	1638
 SmartSweepSettlingTime	1639
 SmartSweepShowIterations	1640
 SmartSweepTolerance	1641
 Smoothing Aperture	1642
 Smoothing ON/OFF	1643
 SnPFormat	1644
 SoftwareGateState Property	1645
 Sound On Fail	1646
 Source	1647
 Sources Property	1648

SourceAttenuator Property	1649
SourceAttenuator PR Property	1650
SourceCount	1651
SourceImpedance Property	1652
Source Port	1653
SourcePortCount	1654
SourcePortMode	1655
SourcePortNames	1657
SourcePowerCalPowerOffset	1659
Source Power Correction	1660
Source Power Option	1661
Source Power State	1662
SourcePullForSParameters Property	1663
SourceStepAttenuatorStepSize Property	1664
Span	1665
Span (Meas)	1666
SParameterCalPorts Property	1667
SpectrumCenterFrequency	1668
SpectrumSpanFrequency	1669
SpectrumStartFrequency	1670
SpectrumStopFrequency	1671
SSB_BalPortNegative	1672
SSB_BalPortPositive	1673
SSB_SEPort1	1674
SSB_SEPort2	1675
SSBMeasurement	1676
Stage1Coefficients	1677
Stage1Frequency	1678
Stage1MaximumCoefficient	1679
Stage1MaximumCoefficientCount	1680
Stage1MaximumCoefficientSum	1681
Stage1MinimumCoefficientCount	1682
Stage2Coefficients	1683

 Stage2MaximumCoefficient	1684
 Stage2MaximumCoefficientCount	1685
 Stage2MaximumCoefficientSum	1686
 Stage2MinimumCoefficientCount	1687
 Stage3FilterType Property	1688
 Stage3FilterTypes Property	1689
 Stage3Parameter Property	1690
 Stage3ParameterMaximum Property	1692
 Stage3ParameterMinimum Property	1693
 Stage3Parameters Property	1694
 Standard Deviation	1695
 Standard For Class	1696
 Start Frequency_CS	1698
 Start Frequency	1699
 Start Power	1701
 Start	1702
 Start (Meas)	1703
 Start DC Property	1704
 StartFrequency PR Property	1705
 StartPhase Property	1706
 StartPowerEx	1707
 StartRatioedPower Property	1708
 State GPL Property	1709
 State	1710
 State DC Property	1713
 State rx	1715
 State pulse	1716
 Statistics Range	1717
 StatusAsString	1718
 Step Rise Time	1719
 StepData	1720
 StepTitle	1721
 StimulusValues	1722

Stop DC Property	1723
Stop Frequency	1724
Stop Frequency_CS	1726
Stop Power	1727
Stop	1728
Stop (Meas)	1729
StopPhase Property	1730
StopPowerEx	1731
StopRatioedPower Property	1732
strPort2Pdeembed_S2PFile	1733
strPortMatch_S2PFile	1734
SubPointTrigger Property	1735
SupportedGenericParameters Property	1736
SupportedParameters Property	1737
Sweep Delay Property	1738
SweepDwell Property	1739
SweepEndMode	1740
SweepHoldOff	1741
SweepOrder	1742
SweepSpeedMode	1743
Sweep Generation Mode	1744
Sweep Time	1745
SweepTimeOption	1746
SweepType (imd)	1747
SweepType (ims)	1749
Sweep Type	1750
System Impedance Z0	1752
SystemName	1753
Target Value	1754
Test Port Power	1755
TestSetType	1757
Text	1758
ThruCalMethod (FCA)	1759

 ThruCalMethod	1760
 ThruPortList	1762
 TimeOut Property	1764
 Title	1765
 Title State	1766
 Tolerance	1767
 TonePower	1768
 TonePowerSetAt Property	1769
 TonePowerStart	1770
 TonePowerStop	1771
 TotalIterations	1772
 TotalNumberOfPoints	1773
 Touchscreen	1774
 Trace Math	1775
 TraceMax	1776
 TraceTitle	1777
 TraceTitleState	1778
 Tracking	1779
 TrackingChannel	1780
 TrackingEnable	1781
 TrackingManualStepEnable	1782
 TrackingStepIndex	1783
 Transform Mode	1366
 Trigger Delay	1784
 TriggerInPolarity	1785
 TriggerInType	1787
 TriggerMode ExtDev Property	1789
 TriggerOutDuration	1790
 TriggerOutInterval	1791
 TriggerOutPolarity	1792
 TriggerOutPosition	1793
 TriggerOutputEnabled	1794
 TriggerPort Property	1795

 TuningIFBW	1796
 TuningMode	1797
 TuningSweepInterval	1798
 TwoPointGroupDelayAperture Property	1799
 Trigger Mode	1800
 Trigger Signal	1802
 Trigger Type	1804
 Type (calstd)	1805
 Type_ts	1806
 TZImag Property	1807
 TZReal Property	1808
 UnusedChannelNumbers	1809
 USBPowerMeterCatalog	1810
 UseCalWindow	1811
 UsedChannelNumbers	1812
 UseMultipleSensors Property	1813
 Use Power Loss Segments	1814
 Use Power Sensor Frequency Limits	1815
 UserCalsetPrefix Property	1817
 User Range	1818
 User Range Max	1819
 User Range Min	1821
 UserDescriptionOfPNA	1823
 UserName	1824
 UserPresetEnable	1825
 Valid	1826
 ValidConnectorType	1827
 Value element	1829
 Values	1830
 Velocity Factor	1831
 View	1832
 Visible	1833
 WGCutoffFreq	1834

WideBandDectionState Property	1835
Width	1836
Window Number	1837
Window State	1838
XAxisAnnotation	1839
XAxis Point Spacing	1840
XAxisStart	1841
XAxisStop	1842
YAxisAnnotation	1843
YScale	1844
Z0	1845
Methods	
Abort	1846
AbortPowerAcquisition	1847
Acquire Cal Standard	1848
Acquire Cal Standard2	1850
AcquireCalConfidenceCheckECALEx	1852
AcquirePowerReadingsEx	1854
AcquireStep	1856
Activate	1858
Activate Marker	1859
Activate Window	1860
Add (channels)	1861
Add (measurement)	1862
Add (naWindows)	1866
Add (PowerLossSegment)	1867
Add (PowerSensorCalFactorSegment)	1868
Add (segments)	1869
Add External Device Method	1870
Add GuidedPowerSensors Method	1871
Add Testset	1872
AddSegment Method	1873
Allow All Events	1874

 AllowChannelToSweepDuringCalAcquisition	1875
 Allow Event Category	1877
 Allow Event Message	1878
 Allow Event Severity	1879
 Apply	1880
 ApplyDeltaMatchFromCalSet	1881
 ApplyPowerCorrectionValuesEx	1882
 ApplySourcePowerCorrectionTo	1884
 AssignSourceToRole	1885
 AutoOrient Method	1887
 AutoPortExtMeasure	1889
 AutoPortExtReset	1890
 Autoscale	1891
 Averaging Restart	1892
 Build Hybrid Kit	1893
 Calculate Error Coefficients	1895
 Calculate	1896
 Calculate cv Method	1898
 Change Parameter	1900
 CheckPower	1904
 Clear	1906
 Clear fifo	1907
 Close CalSet	1908
 ComputeErrorTerms	1909
 ConfigEnhancedNB2	1910
 ConfigEnhancedNBIFAtten	1913
 ConfigNarrowBand3	1914
 ConfigurationFile	1917
 Configurations	1918
 Configure	1919
 Continuous Sweep	1920
 Copy	1921
 CopyFrom Method	1923

 CopyToChannel	1924
 Create SParameter Method	1925
 CreateCalSet	1926
 CreateCustomCal	1927
 CreateCustomCalEx	1928
 CreateCustomMeasurementEx	1930
 CustomCalConfiguration Method	1016
 Create Measurement	1936
 DataToMemory	1940
 Deembed Method	1941
 Delete	1943
 Delete Marker	1944
 Delete All Markers	1945
 DeleteAllSegments Method	1946
 DeleteCalSet	1947
 DeleteConfiguration	1948
 DeleteSegment Method	1949
 Delete ShortCut	1950
 Delta Marker	1039
 Disallow All Events	1951
 DiscardChanges	1952
 DisplayNAWindowDuringCalAcquisition	1953
 DisplayOnlyCalWindowDuringCalAcquisition	1955
 Do Print	1956
 DoECAL1PortEx	1957
 DoECAL2PortEx	1959
 DoneCalConfidenceCheckECAL	1961
 DoReceiverPowerCal	1962
 DoResponseCal Method	1964
 Embed Method	1965
 EnumerateCalSets	1967
 EnumerateItems	1968
 Execute	1969

 Execute Shortcut	1970
 Exists Method	1971
 GenerateGlobalDeltaMatchSequence	1972
 GenerateErrorTerms	1973
 GenerateSteps	1974
 GetAllSegments	1976
 Get AuxIO	1978
 Get Cal Standard	1979
 GetCalKitTypeString Method	1980
 GetCompatibleCalKits Method	1981
 Get CalManager	1983
 Get CalSetByGUID	1984
 Get CalSetCatalog	1985
 Get CalSetUsageInfo	1986
 Get Cal Types	1988
 Get Complex	1989
 GetConnectedPhaseReferences Method	1991
 GetConverter	1992
 Get DataByString	1993
 Get Data	1996
 Get ECALModuleInfoEx	1999
 GetEcalUserCharacterizer	2000
 Get ENRData	2001
 Get ErrorCorrection	2002
 Get Error Term	2003
 Get Error Term2	2005
 Get Error Term By String	2007
 Get Error Term Complex	2009
 Get Error Term Complex2	2011
 Get Error Term Complex By String	2013
 Get Error Term List	2015
 Get Error Term List2	2017
 GetErrorTermStimulus Method	2018

 Get ExtendedCallInterface	2020
 Get ExternalTestSetIO	2021
 Get Filter Statistics	2022
 Get Guid	2023
 get InputVoltageEX	2024
 Get Input1	2026
 GetIPConfigurationStruct	2027
 Get IsolationPaths	2029
 GetLibraryFunctions Method	2030
 Get MaterialHandlerIO	2031
 Get NAComplex	2032
 Get NumberOfGroups	2034
 Get Output	2035
 Get Output Voltage	2036
 Get OutputVoltage Mode	2037
 Get Paired Data	2038
 Get Port	2040
 Get PortC Data	2041
 Get PortNumber	2042
 GetRaw2DDData	2043
 GetRaw2DDDataIm	2046
 GetRaw2DDDataRe	2048
 GetRxlLevelingConfiguration	2050
 GetSourceByRole	2051
 GetSourceRoles	2052
 Get Reference Marker	2053
 Get Required Eterm Names	2054
 Get Scalar	2055
 Get Shortcut	2057
 Get SnPData	2058
 Get SnpDataWithSpecifiedPorts	2060
 Get SourcePowerCalDataEx	2062
 Get SourcePowerCalDataScalarEx	2064

 LoadTheme	2110
 Manual Trigger	2111
 Move	2112
 NetworkPortMap	2113
 Next IF Bandwidth	2114
 Number of Groups	2115
 Open CalSet	2116
 Parse	2118
 Preset (app and chan)	2119
 Previous IF Bandwidth	2120
 Print To File	2121
 Put Complex	2122
 Put Data Complex	2124
 PutENRData	2126
 Put ErrorTerm	2127
 Put ErrorTerm2	2129
 Put Error Term By String	2131
 Put ErrorTerm Complex	2132
 Put ErrorTerm Complex2	2134
 Put Error Term Complex By String	2136
 PutErrorTermStimulus Method	2137
 Put Formatted Scalar Data	2139
 Put NAComplex	2141
 Put Output	2144
 Put Output Voltage	2145
 Put Output Voltage Mode	2146
 Put Port	2147
 Put PortCData	2149
 Put Scalar	2150
 Put Shortcut	2152
 Put SourcePowerCalDataEx	2153
 Put SourcePowerCalDataScalarEx	2155
 Put Standard	2157

 Put Standard By String	2159
 Put Standard Complex	2160
 Put Standard Complex By String	2162
 Quit	2163
 Read Data	2164
 Read Raw	2165
 ReCalculate Method	2167
 Recall	2168
 Recall Kits	2169
 Remove	2170
 RemoveAll Method	2172
 RemoveChannelNumber	2173
 RemoveItem	2174
 RemoveLibrary Method	2175
 Reset	2176
 Reset (CalAll) Method	2177
 Reset (PhaseRef) Method	2178
 ResetLOFrequency	2179
 ResetTheme Method	2180
 ResetTuningParameters	2181
 Restore Cal Kit Defaults	2182
 Restore Cal Kit Defaults All	2183
 Restore Defaults	2184
 Resume	2185
 Save	2186
 Save (CalSet)	2188
 Save CalSets	2189
 SaveCitiDataData	2190
 SaveCitiFormattedData	2191
 SaveData Method	2192
 SaveENRFile	2196
 Save File	2197
 SaveFile ED Method	2198

 SaveToDiskMemory Method	2199
 SaveToECal	2200
 Save Kits	2201
 SearchCompressionPoint	2202
 SearchPowerNormalOperatingPoint Method	2203
 SearchPowerSaturation Method	2204
 Search Filter Bandwidth	2205
 Search Max	2206
 Search Min	2207
 Search Next Peak	2208
 Search Peak Left	2209
 Search Peak Right	2210
 Search Target	2211
 Search Target Left	2212
 Search Target Right	2213
 SegmentCalculate Method	2214
 SelectCalSet	2216
 Set All Segments	2218
 Set BBPorts	2221
 SetBSPorts Method	2222
 Set Cal Info	2223
 SetCallInfoEx	2225
 Set Center	2227
 Set CW	2228
 SetCWFreq	2229
 SetDutPorts	2230
 Set Electrical Delay	2231
 Set FailOnOverRange	2232
 Set IsolationPaths	2233
 SetIPConfiguration	2235
 Set PowerAcquisitionDevice	2237
 Set Frequency LowPass	2238
 SetPortMap	2239

 Set Reference Level	2241
 Set SBPorts	2242
 Set SSBPorts	2243
 SetupMeasurementsForStep	2244
 Set StandardsForClass	2245
 Set Start	2247
 Set Stop	2248
 Show Marker Readout	2249
 Show Status Bar	2250
 Show Stimulus	2251
 Show Table	2252
 Show Title Bars	2253
 Show Toolbar	2254
 Single	2256
 Store	2258
 StoreConfiguration	2259
 StoreTheme Method	2260
 StringToNACalClass	2261
 StringtoNAErrorTerm2	2263
 SweepOnlyCalChannelDuringCalAcquisition	2264
 TestsetCatalog	2265
 UserPreset	2266
 UserPresetLoadFile	2267
 UserPresetSaveState	2268
 Write Data	2269
 Write Raw	2270
 WriteSnpFileWithSpecifiedPorts	2272
 Events	
 OnCalEvent	2274
 OnChannelEvent	2277
 OnDisplayEvent	2279
 OnHardwareEvent	2281
 OnMeasurementEvent	2283

OnSCPIEvent	2285
OnSystemEvent	2287
OnUserEvent	2289
Examples	
C Example	2290
CalSet_Examples	2295
Getting Trace Data from the Analyzer	2297
Perform a Guided Cal using COM	2300
Perform a Source Power Cal	2304
Perform an Unguided Cal using COM	2307
Perform an Unknown Thru or TRL Cal	2310
Perform Global Delta Match Cal	2312
Perform a Guided Cal with CSharp	2313
Perform an ECal	2318
Perform an ECal User Characterization	2321
Perform a Comprehensive Guided 2-Port Cal	2327
Perform an ECAL Confidence Check	2334
Writing Cal Set Data using COM	2337
Upload a Source Power Cal	2339
Upload Segment Table	2342
Create and Cal an SMC Measurement	2347
Create and Cal a VMC Measurement	2351
Create an SMC Fixed Output Meas	2356
Create a Segmented Sweep for Mixers	2359
Use Existing Power Cal for SMC	2363
Create a Balanced Measurement	2366
Create a PMAR Device and Measurement	2370
Create a Wideband Pulsed Measurement using the PNA-X	2373
Create an IM Spectrum Measurement	2376
Create an iTMSA Measurement	2379
Create_and_Cal_a_Gain_Compression_Measurement	2381
Create and Cal a GCX Measurement	2388
Create and Cal a NoiseFigure Measurement	2393

 Create and Cal an NFX Measurement	2398
 Create and Cal an IMD Measurement	2404
 ENR File Management Example	2408
 Events with C	2410
 FOM Examples	2413
 Limit Line Testing Example with COM	2416
 Modify Display Colors	2417
 E5091 Testset Control	2419
 Errors and the SCPIStringParser Object	2420
 External Testset Control	2423
 PathConfiguration Example	2425
 PNA-X Create a Pulsed Measurement	2428
 Setup Basic Measurements	2433
 Setup Compression Marker	2435
 Set Up Embedded LO Measurement	2437
 Setup FastCW and FIFO	2439
 Setup Noise Figure Port Mapping	2441
 Setup Phase Control	2443
 Setup PNOP and PSAT Marker Search	2445
 Setup Receiver Leving	2448
 Show Custom Window during Calibration	2450
 Events Example	2453
 Upload_Segment_Table_in_C	2455
 Using C#	2464
 Concepts	
 Configure for COM-DCOM Programming	2467
 COM Fundamentals	2472
 Getting a Handle to an Object	2476
 Collections in the Analyzer	2480
 COM Data Types	2482
 PNA Automation Interfaces	2485
 Working with the Analyzer's Events	2487
 Read and Write Calibration Data using COM	2492

C and the COM Interface	2494
Using .NET	2498
SCPI	
Commands	
SCPI Command Tree	2500
Common Commands	2503
Abort	2507
Calculate	
Correction	2508
Custom	2517
Data	2523
Equation	2532
Filter	2537
Format	2543
FSimulator	2546
Function	2548
GCDATA	2553
GCMeas	2558
GroupDelay	2562
Limit	2565
Marker	2573
Math	2594
Mixer	2596
Normalize	2597
Offset	2600
Parameter	2603
RData	2613
Smoothing	2615
Transform	2618
X Values	2627
Calpod	2628
Control	2634
CSET	2646

Display	2652
Format	2681
Hardcopy	2684
Initiate	2697
Memory	2700
Output	2715
Route	2717
Sense	
Average	2718
Bandwidth	2721
Class	2723
Correction	2724
Cal Kit	2753
Cal Stds	2767
Cal Sets	2791
Extensions	2808
Guided Cal	2822
IMD	2852
Session	2858
SMC	2864
VMC	2876
Couple	2887
FOM	2889
FOMSegment	2899
Frequency	2909
Gain Compression	2913
IF (PNA-X)	2929
IMD	2939
IMS	2961
Mixer	2976
MixerEmbedLO	2999
Multiplexer	3010
Noise	3021

Offset	3036
Path	3040
Power	3046
Pulse	3047
Roscillator	3057
Segment	3058
Sweep	3073
XAxis	3084
Source	3085
Source Correction	3099
Status	3119
System	3135
CalAll	3159
CalPhase	3168
ConfigExtDevice	3175
ConfigExtDC	3185
ConfigExtPMAR	3190
ConfigExtPulseGen	3199
CorrIntLinear	3203
FIFO	3206
Preferences	3209
Trigger	3218
Examples	
Catalog Measurements using SCPI	3232
Channels, Windows, and Measurements using SCPI	3233
Controll,Talk,Listen using SCPI	3235
Create a Balanced Measurement using SCPI	3237
Create a measurement using SCPI	3243
Create a Narrowband Point-in-Pulse Measurement_PNA-X	3244
Create a Narrowband Pulse Profile Measurement - PNA-X	3250
Create a PMAR Device and Measurement	3257
Create a Swept IMDX Measurement	3259
Create a Wideband Pulse Measurement	3263

Create an FOM Measurement	3266
Create an iTMSA Measurement	3270
Create an SMC Fixed Output Meas	3272
Create_and_Cal_a_GCA_Measurement	3276
Create and Cal a GCX Measurement	3284
Create and Cal a Noise Figure Measurement	3292
Create and Cal a VMC Measurement	3297
Create and Cal an IMD Measurement	3302
Create and Cal an NFX Measurement	3307
Create and Cal an SMC Measurement	3314
Create and Cal Multiple SMC Channels	3319
Create New Cal Kit using SCPI	3322
Custom Power Meter Driver	3329
ECALConfidence Check using SCPI	3333
Establish a VISA Session	3336
External Test Set using SCPI wLink	3338
Getting and Putting Data using SCPI	3340
GPIB Pass Through	3342
GPIB using Visual C	3344
Load Eterms into Cal Sequence	3348
Modify a Calibration Kit using SCPI	3349
Guided 2-Port or 4-Port Cal	3351
Perform a Simple Source Power Cal	3355
Perform a Source and Receiver Power Cal using SCPI	3357
Perform a Source Power Cal with TWO Sensors	3368
Perform an ECal User Characterization	3370
Perform an Unguided Cal on a 4-Port PNA using SCPI	3379
Perform Global Delta Match Cal	3389
Perform Guided 1Port	3390
Perform Guided 2-Port Comprehensive Cal	3392
Perform Guided ECal	3401
Perform Guided Mechanical Cal	3403
Perform Guided TRL Calibration	3405

Perform an Unguided 1-Port Cal on Port 2	3408
Perform Unguided 2-Port Mech Cal	3410
Perform Unguided ECAL	3412
Perform Unknown Thru or TRL Cal	3414
Setup FastCW and FIFO	3417
Setup Markers	3421
Setup Noise Figure Port Mapping	3423
Setup Phase Control	3425
Setup PNOP and PSAT Markers	3427
Setup RxLeveling	3430
Setup Sweep Parameters using SCPI	3432
Setup the Display using SCPI	3433
Show Custom Window during Calibration	3435
Sliding Load Cal using SCPI	3439
Socket Client	3440
Status Reporting using SCPI	3449
Transfer Data using GPIB	3451
Triggering the PNA using SCPI	3453
Unguided_Cal_on_Multiple_Channels	3460
Upload and Download a Segment List	3465
Uploading a Source Power Cal using SCPI	3470
Concepts	
GP-IB Fundamentals	3474
The Rules and Syntax of SCPI Commands	3479
How to Configure for GPIB, SCPI, and SIDL	3484
Getting Data from the Analyzer	3488
Understanding Command Synchronization	3493
Calibrating the PNA Using SCPI	3500
The PNA as a USB Device	3506
Reading the Analyzer's Status Registers	3508
Referring to Traces, Measurements, Channels, and Windows Using SCPI	3513
Remote Control of SCPI USB Devices Connected to a PNA	3515

Configure for VISA and SIDL	3517
VEE Examples	
VEE Pro Runtime	3521
Basic Control VEE	3522
ECal with Confidence Check using VEE	3524
Data Access Map	
DataMapSet	3525
Rear Panel IO Connectors	
Interface Control	3526
External TestSet IO Connector	3532
Material Handler IO Connector	3539
Pulse IO (PNA-X)	3553
Power IO (PNA-X)	3555
Programming Guide	579
New Programming Commands	3558
COM versus SCPI	3587
Remotely Specifying a Source Port	3590
Shut Down the PNA Remotely	3592
LXI Compliance	92
Using Macros	266
Applications	
Antenna Features	3593
Frequency Converter Application (Opts 082 / 083)	
FCA Overview	3598
SMC Measurements	3613
SMC plus Phase	3630
SMC Phase Reference Calibration	3636
How to make an SMC Fixed Output Measurement	3646
SMC with a Booster Amp	3653
VMC Measurements	3656
How to make a VMC Fixed Output Measurement	3674
Embedded LO (Opt 084)	3682
Characterize Adaptor Macro	422
Frequency Offset (Opt 080)	

Frequency Offset (Option 080)	3689
Frequency Converting Device Measurements	3700
Frequency Offset Calibration	3701
Conversion Loss	3704
Conversion Compression	3708
Isolation	3711
Harmonic Distortion	3715
Return Loss and VSWR	3717
Gain Compression Application (Opt 086)	3719
Gain Compression for Converters	3747
Gain Compression Cal	3756
Integrated Pulse Application (Opt 008)	3760
iTMSA (Opt 460)	3770
Noise Figure	
Noise Figure Application	3782
Noise Figure on Converters	3810
Noise Figure Cal	3822
Noise Figure and TRL Cal	3832
Pulsed Application (Opt H08)	3836
Swept IMD and IM Spectrum (Opt 087)	
Swept IMD and IM Spectrum Concepts	3852
Swept IMD	3862
IM Spectrum	3879
Swept IMDx for Converters	3888
IMx Spectrum for Converters	3900
Time Domain (Opt 010)	3909
WideBand Pulsed App	3928
Networking the PNA	
Drive Mapping	572
Product Support	
Troubleshoot the PNA	3929
List of Error Messages	3933
About Error Messages	4016

Accessories	4020
USB to GPIB Adapter	4027
Firmware Update	4029
PNA Configurations and Options	4035
Option Enable	4045
Instrument Calibration	4049
Other Resources	4051
SCPI Errors	4052
Technical Support	4063
Diagnostic Tools and Adjustments	
10 MHz Reference Adjust	4069
Display Test	4071
Operators Check	4072
Phase-Lock IF Gain Adjust	4076
System Verification	4079
Source Adjustment	4090
Receiver Cal	4092
Receiver Display	4095
IF Access	
IF Access User Interface Settings	4096
External Test Head Configuration	4106
PNA-X IF Path Configuration	4096
N5264A Measurement Receiver	4120
System Settings	
Configure External Devices	4124
Configure an External Device	4124
Configure an External Source	4130
Configure a DC Device	4136
Configure a Power Meter As Receiver	4144
Configure an External Pulse Generator	4155
Synchronize an External PSG Source	4160
E5091 Test Set Control	4165
Error Messages	4016

External Testset Control	4169
Display Colors	4182
Frequency Blanking	78
Interface Control	3526
Mechanical Devices	4186
PNA Preferences	84
Power Limit and Power Offset	4189
System Impedance	4192
Rear Panel IO Connectors	
Interface Control	3526
External TestSet IO Connector	3532
Material Handler IO Connector	3539
Pulse IO (PNA-X)	3553
Power IO (PNA-X)	3555
Tutorials	
App Notes	4194
Network Analyzer Basics	4198
Connector Care	4199
ESD Protection	4210
Measurements	
Absolute Output Power	4211
Active Probing	4214
AM-PM Conversion	4216
Amplifier Measurements	4221
Antenna Measurements	4224
Balanced Measurements	4227
Complex Impedance	4236
Comparing the PNA Delay Functions	4239
Deviation from Linear Phase	4241
External Source Control	4160
Gain and Flatness	4244
Gain Compression	4247
Group Delay	4252

High-Gain Amplifier Measurements	4259
High Power Measurements using a PNA-X	4261
Phase Measurements	4263
Reverse Isolation	4267
Reflection Measurements	4270
Time Domain Measurements	3909
Specifications	4275
Glossary	4277

What's New in PNA Code Version A.10.00

- [Windows® Embedded Standard 7](#)
- [HiSLiP](#)
- [Integrated CalKit Editor](#)
- [Increased Number of Points to 100K](#)
- [Cal Kits saved and recalled using *.xkt format](#)
- [U8480 Power Sensors Supported](#)
- [Mechanical Devices: Copy Active Channel](#)
- [Search Within \(for Markers\)](#)
- [Dialog Transparency](#)
- [Move App to Back](#)
- [System time on PNA Status Bar](#)
- [Right-click menus on Status Bar](#)
- [Elimination of 3 Toolbars](#)
- [Enhanced MATLAB® functionality](#)

Notes

- This version includes changes from all versions listed below.
- See [new A.10.00 Programming Commands](#)

To see the PNA code version that is currently installed on the PNA, click **Help**, then **About Network Analyzer**

What's New in PNA Code Version A.09.90

- [SMC Phase Reference Cal from 10 MHz](#)
- [CalPod as ECal](#)
- [External SMU Device](#)
- Equation Editor enhancements

- [New marker functions](#)
 - [Use Short Names](#)
-

What's New in PNA Code Version A.09.85

- [Support for U2020 X-Series USB Power Sensors](#)
- [Delta Match Calibration](#) required on ALL N5231A, N5232A, and N5239A models.
- [Unguided TRL Cal](#) allowed on all PNA 4-port models
- [Noise Parameters](#)

What's New in PNA Code Version A.09.80

- **Standard Measurements**
 - [Calibrate All Channels](#)
 - [SE => Balanced Topology](#)
 - [Increased Segments in Power Sensor Loss Table](#)
 - [EXG Sources supported](#)
 - [CalPod](#)
 - **Application Enhancements**
 - [SMC with Phase Reference](#)
 - [Noise Receiver Cal with Power Sensor](#)
 - [PNA models with 50 GHz Noise Receivers](#)
-

What's New in PNA Code Version A.09.60

- New [N523xA models](#)
-

What's New in PNA Code Version A.09.50

Standard Measurements

- [Unguided Cals can access 95 Cal Kits](#)

- [Preference setting to list Recall files](#)
- [YouTube Videos](#)
- [Display Menu changes](#)
- [Undo/Redo](#)
- [Quick Start](#)
- [Quickly change Scale, Reference Level, and Position](#)
- [Redesigned Receiver Leveling dialog](#)
- New External Device configuration:
 - [Pulse Generators](#)
 - [DC Sources and Meters](#)
 - [DC Source Control dialog](#)

Application Enhancements

- [External Pulse Generator in Integrated Pulse App](#)
 - [IMoSpectrum in mmWave](#)
-

What's New in PNA Code Version A.09.42

May 2013: [Support for U2020X Power Sensors](#)

- [New N522x Models](#)

Standard Measurements - available on all Models / Options

- [Drag a trace to another window](#)

Application Enhancements

- [Copy Channels on all Applications](#)
- ["Src 2 out Port 2" factory configuration on PNA-X Opt 423.](#)
- [IMD f2 Tone using External Source / Combiner](#)

- [IMD and IM Spectrum Tone Power Leveling settings](#)
 - [IMD, IMDx, and IM Spectrum "Min" and "Max" parameters](#)
 - [Use a Power Table with mmWave SMC Measurements](#)
 - [mmWave; Mixer mode - 2-port test set on 4-port PNA](#)
 - [Guided Power Cal for SMC](#)
 - [ESG and PSG Sources for Phase Control](#)
-

What's New in PNA Code Version A.09.33

New Options

- [Source Phase Control](#) - Opt 088

Application Enhancements

- [FCA Update](#) - Opt 082, and 083

Standard Measurements - available on all Models / Options

- [Security for External Sources](#)
- [2-Port](#) and [4-Port](#) Fixture Extrapolation and Reverse Ports.
- [Phase Coherent "R over R" measurements](#)
- [Use Multiple Power Sensors for Guided Power Cal](#)
- [Perform Source Power Cal with PMAR Device](#)

Tip - Do you access the same PNA dialog often?

Your Favorites are always two keystrokes away. [Learn how.](#)

What's New in PNA Code Version A.09.31

- [Support for N1913A and N1914A Power Meters](#)
-

What's New in PNA Code Version A.09.30

- [N5247A - 67 GHz PNA-X](#)

Application Enhancements

- [Gain Compression on Converters \(GCX\)](#)
- [Support for Dual-Stage Converters in all Apps](#)

Standard Measurements - available on all Models / Options

- [Enhanced S-parameter Power Cal](#)
 - [Marker Display enhancements](#)
 - [Perform Source Power Cal at multiple power levels](#)
 - [IF Gain Setting](#)
 - [Receiver Overload/Compression Warning](#) and [Power OFF](#) Preferences
 - [Confirm changes on Meas Class dialog](#)
 - [DSP Version 5](#)
-

What's New in PNA Code Version A.09.22

- [Use any PNA-X Ports with Noise Figure Opt 028.](#)
-

What's New in PNA Code Version A.09.20

Application Enhancements

- [Integrated Pulse Measurements](#) (Opt 008)
- [Noise Figure using Standard PNA Receiver](#) (Opt 028)
- [Noise Figure on N5244A/45A](#) (Opt H29)
- [Edge and Level Trigger in Pulse](#)
- [Exclude SC12 Sweep for SMC \(Opt 082/083\)](#)
- [Include Phase with SMC](#) (PNA-X with Opt 083)
- [Fixturing in Apps](#)
- [Max Output Power for GCA](#) (Opt 086)

Standard Measurements - available on all Models / Options

- [Mechanical Device conflicts cause Channel Block \(NOT Channel Hold\)](#)
 - [PSAT Marker and Power Normal Operating Point Marker](#)
 - [Group Delay Aperture Setting](#)
 - [Active Background Display Color](#)
 - [Solid or Dotted Grid Lines](#)
 - [Point Sweep on PNA "C" Models](#)
 - [Fixture Power Compensation](#)
 - [Sweep Delay](#)
 - [Uncertainty equations using RSS Computations](#)
 - [Preset Power Preference Setting](#)
 - [Use Last Receiver Leveling Correction for SPC](#)
 - Data Save Enhancements
 - [Recall .SNP files to view as trace](#)
 - ["Save Data As" Dialog](#)
 - [Save Balanced Data as SNP files](#)
 - Characterize Adaptor Macro Rev. A.02.10
 - [Reverse S2P](#)
 - [Load the PNA Power Loss Table from an existing S2P file](#)
-

What's New in PNA Code Version A.09.10

- [Noise Figure on Converters \(NFX\)](#)
 - [AgileUpdate for Customer Releases](#)
 - [New Help, About Capabilities](#)
-

What's New in PNA Code Version A.09.00

- [External Device Configuration](#)

- [Power Meter as Receiver \(PMAR\)](#)
- [Power Offsets and Limits](#)
- [Display and Print Colors](#)
- [Scale Coupling](#)
- [Mechanical Device Settings](#)
- [Device side USB](#)
- [Increased Number of Channels to 200](#)
- [ECal User Chars Saved to PNA Disk Memory](#)

Application Enhancements

- [GCA Compression Analysis](#)
- [GCA Compression from Saturation](#)
- [Receiver Leveling on IMD, FCA, and GCA](#)
- [Embedded LO on SMC and IMDx](#)
- [Limited Port Mapping on IMD](#)
- [Point Averaging on FCA and IMD](#)
- [SMC Measurements with mmWave Modules](#)
- [mmWave Measurements with no Test Set](#)

What's New in PNA Code Version A.08.60

- [New 40 GHz and 50 GHz PNA-X Models](#)

What's New in PNA Code Version A.08.55

- [IMDx \(Swept and Spectrum\) for Converters](#)
 - [ADC Measurements in a Gain Compression channel.](#)
 - [New 13.5 GHz PNA-X Model](#)
-

What's New in PNA Code Version A.08.50

- [Fast Antenna Features for the PNA-X](#)
 - [Receiver Leveling](#)
 - [Phase Sweep](#) in iTMSA
 - [Up to 25 User Macros](#)
 - [Gain Compression Marker](#)
 - [Port Extensions enhancements](#)
 - [Electrical Delay enhancements](#)
 - [Extra Security enhancement](#)
 - Save [*.CSV](#) and [*.MDF](#) File Types
 - Noise Figure App enhancements:
 - [Scalar Noise Figure measurement](#)
 - [Incident Noise Power parameters](#)
 - MM Module enhancements:
 - [Power Level Control](#)
 - Supports [iTMSA](#)
 - [Max point count to 32,001](#)
 - [Faster Power Sweeps](#)
-

What's New in PNA Code Version A.08.35

- [New N5264A model](#)
-

What's New in PNA Code Version A.08.33

- [IMD Application](#) (Opt 087)
- [Fast Sweep Mode](#)
- [New Equation Editor functions](#)

- [20001 Segments in Power Loss Table](#)
 - [Data Format Units](#)
 - [Marker=>CW Freq Function](#)
 - [Up to 12 User Characterizations](#)
 - [Characterize Adaptor Macro 2.0](#)
-

What's New in PNA Code Version A.08.20

- [iTMSA](#)
 - [User Preferences dialog](#)
 - [Uncoupled Power Sweep](#)
 - [Equation Editor Import Functions](#)
 - [Wider Traces Preference](#)
 - [24 Traces per Window](#)
 - [LXI Compliance](#)
 - [GCA Enhancements](#)
 - [FCA - selectable ports](#)
 - [Support for N5261A and N5262A MM test sets](#)
 - [cXL Code Translation Software - Quick Link](#)
-

What's New in PNA Code Version A.08.00

- [Noise Figure Application \(Opt 029\)](#)
- [Gain Compression Application \(Opt 086\)](#)
- ['Sweep' Trigger Mode](#)
- [Custom Cal Window settings](#) (remote only)
- [New Equation Editor Functions](#)
- [Minimum Number of Points = 1](#)

MICROSOFT SOFTWARE LICENSE TERMS

WINDOWS EMBEDDED STANDARD 7

These license terms are an agreement between you and Agilent Technologies. Please read them. They apply to the software included on this device. The software also includes any separate media on which you received the software.

The software on this device includes software licensed from Microsoft Corporation or its affiliate.

The terms also apply to any Microsoft

- updates,
- supplements,
- Internet-based services, and
- support services

for this software, unless other terms accompany those items. If so, those terms apply.

If you obtain updates or supplements directly from Microsoft, then Microsoft, and not Agilent Technologies, licenses those to you.

As described below, using the software also operates as your consent to the transmission of certain computer information for Internet-based services.

By using the software, you accept these terms. If you do not accept them, do not use the software. Instead, contact Agilent Technologies to determine its return policy for a refund or credit.

If you comply with these license terms, you have the rights below.

1. USE RIGHTS

Use. The software license is permanently assigned to the device with which you acquired the software. You may use the software on the device.

2. ADDITIONAL LICENSING REQUIREMENTS AND/OR USE RIGHTS

a. Specific Use. Agilent Technologies designed the device for a specific use. You may only use the software for that use.

b. Other Software. You may use other programs with the software as long as the other programs

- directly supports the manufacturers specific use for the device, or
- provide system utilities, resource management, or anti-virus or similar protection.

" Software that provides consumer or business tasks or processes may not be run on the device. This includes email, word processing, spreadsheet, database, scheduling and personal finance software. The device may use terminal services protocols to access such software running on a server.

c. Device Connections. You may not use the software as server software. In other words, more than one device may not access, display, run, share or use the software at the same time.

You may use terminal services protocols to connect the device to a server running business task or processes software such as email, word processing, scheduling or spreadsheets.

You may allow up to ten other devices to access the software to use

- File Services,

- Print Services,
- Internet Information Services, and
- Internet Connection Sharing and Telephony Services.

The ten connection limit applies to devices that access the software indirectly through multiplexing or other software or hardware that pools connections. You may use unlimited inbound connections at any time via TCP/IP.

- d. Remote Access Technologies. You may access and use the software remotely from another device using remote access technologies as follows.

Remote Desktop. The single primary user of the device may access a session from any other device using Remote Desktop or similar technologies. A session means the experience of interacting with the software, directly or indirectly, through any combination of input, output and display peripherals. Other users may access a session from any device using these technologies, if the remote device is separately licensed to run the software.

Other Access Technologies. You may use Remote Assistance or similar technologies to share an active session.

Other Remote Uses. You may allow any number of devices to access the software for purposes other than those described in the Device Connections and Remote Access Technologies sections above, such as to synchronize data between devices.

- e. Font Components. While the software is running, you may use its fonts to display and print content. You may only

- embed fonts in content as permitted by the embedding restrictions in the fonts; and
- temporarily download them to a printer or other output device to print content.

- f. Icons, images and sounds. While the software is running, you may use but not share its icons, images, sounds, and media.

3. VHD BOOT. Additional copies of the software created using the software's Virtual Hard Disk functionality (VHD Image) may be pre-installed on the physical hard disk of the device. These VHD Images may only be used for maintaining or updating the software installed on the physical hard disk or drive. If the VHD Image is the only software on your device, it may be used as the primary operating system but all other copies of the VHD Image may only be used for maintenance and updating.

4. POTENTIALLY UNWANTED SOFTWARE. The software may include Windows Defender. If Windows Defender is turned on, it will search this device for spyware, adware and other potentially unwanted software. If it finds potentially unwanted software, the software will ask you if you want to ignore, disable (quarantine) or remove it. Any potentially unwanted software rated high or severe, will be automatically removed after scanning unless you change the default setting. Removing or disabling potentially unwanted software may result in

- Other software on your device ceasing to work, or
- Your breaching a license to use other software on this device

By using this software, it is possible that you will also remove or disable software that is not potentially unwanted software.

5. SCOPE OF LICENSE. The software is licensed, not sold. This agreement only gives you some rights to use the software. Agilent Technologies and Microsoft reserve all other rights. Unless applicable law gives you more rights despite this limitation, you may use the software only as expressly permitted in this agreement. In doing so, you must comply with any technical limitations in the software that allow you to use it only in certain ways. For more information, see the software documentation or contact Agilent Technologies. You may not:

- " work around any technical limitations in the software;
- " reverse engineer, decompile or disassemble the software;
- " make more copies of the software than specified in this agreement;
- " publish the software for others to copy;
- " rent, lease or lend the software; or
- " use the software for commercial software hosting services.

Except as expressly provided in this agreement, rights to access the software on this device do not give you any right to implement Microsoft patents or other Microsoft intellectual property in software or devices that access this device.

6. INTERNET-BASED SERVICES. Microsoft provides Internet-based services with the software. Microsoft may change or cancel them at any time.

a. Consent for Internet-Based Services. The device may contain one or more of the software features described below. These features connect to Microsoft or service provider computer systems over the Internet. In some cases, you will not receive a separate notice when they connect. For more information about these features, visit go.microsoft.com/fwlink/?linkid=104604.

By using these features, you consent to the transmission of this information. Microsoft does not use the information to identify or contact you.

Computer Information. The following features use Internet protocols, which send to the appropriate systems computer information, such as your Internet protocol address, the type of operating system and browser, the name and version of the software you are using, and the language code of the device where you installed the software. Microsoft uses this information to make the Internet-based services available to you. Agilent Technologies has elected to turn on the following features on the device.

- Plug and Play and Plug and Play Extensions. You may connect new hardware to your device. Your device may not have the drivers needed to communicate with that hardware. If so, the update feature of the software can obtain the correct driver from Microsoft and install it on your device.
- Web Content Features. Features in the software can retrieve related content from Microsoft and provide it to you. Examples of these features are clip art, templates, online training, online assistance and Appshelp. You may choose to switch them off or not use them.
- Digital Certificates. The software uses x.509 version 3 digital certificates. These digital certificates confirm the identity of user sending information to each other and allow you to encrypt the information. The software retrieves certificates and updates certificate revocation lists over the Internet.
- Auto Root Update. The Auto Root Update feature updates the list of trusted certificate authorities. You can switch off this feature.
- Windows Media Digital Rights Management. Content owners use Windows Media digital rights management technology (WMDRM) to protect their intellectual property, including copyrights. This software and third party software use WMDRM to play and copy WMDRM-protected content. If the

software fails to protect the content, content owners may ask Microsoft to revoke the software's ability to use WMDRM to play or copy protected content. Revocation does not affect other content. When you download licenses for protected content, you agree that Microsoft may include a revocation list with the licenses. Content owners may require you to upgrade WMDRM to access their content. Microsoft software that includes WMDRM will ask for your consent prior to the upgrade. If you decline an upgrade, you will not be able to access content that requires the upgrade. You may switch off WMDRM features that access the Internet. When these features are off, you can still play content for which you have a valid license.

- Windows Media Player. When you use Windows Media Player, it checks with Microsoft for
 - " compatible online music services in your region;
 - " new versions of the player; and
 - " codecs if your device does not have the correct ones for playing content.You can switch off this feature. For more information, go to: go.microsoft.com/fwlink/?LinkId=51331.
- Malicious Software Removal/Clean On Upgrade. Before installation of the software, the software will check and remove certain malicious software listed at www.support.microsoft.com/?kbid=890830 (Malware) from your device. When the software checks your device for Malware, a report will be sent to Microsoft about any Malware detected or errors that occurred while the software was checking for Malware. No information that can be used to identify you is included in the report. You may disable the software's Malware reporting functionality by following the instructions found at www.support.microsoft.com/?kbid=890830.
- Network Awareness. This feature determines whether a system is connected to a network by either passive monitoring of network traffic or active DNS or HTTP queries. The query only transfers standard TCP/IP or DNS information for routing purposes. You can switch off the active query feature through a registry setting.
- Windows Time Service. This service synchronizes with www.time.windows.com once a week to provide your device with the correct time. The connection uses standard NTP protocol.
- Search Suggestions Service. In Internet Explorer, when you type a search query in the Instant Search box or type a question mark (?) before your search term in the Address bar, you will see search suggestions as you type (if supported by your search provider). Everything you type in the Instant Search box or in the Address bar when preceded by a question mark (?) is sent to your search provider as you type. Also, when you press Enter or click the Search button, the text in the Instant Search box or Address bar is sent to the search provider. If you use a Microsoft search provider, use of the information sent is subject to the Microsoft Online Privacy Statement. This statement is available at go.microsoft.com/fwlink/?linkid=31493. If you use a third-party search provider, use of the information sent will be subject to the third party's privacy practices. You can turn search suggestions off at any time. To do so, use Manage Add-ons under the Tools button in Internet Explorer. For more information about the search suggestions service, see go.microsoft.com/fwlink/?linkid=128106.
- Consent to Update Infrared Emitter/Receiver. The software may contain technology to ensure the

proper functioning of the infrared emitter/receiver device shipped with certain Media Center-based products. You agree that the software may update the firmware of this device.

- Media Center Online Promotions. If you use Media Center features of the software to access Internet-based content or other Internet-based services, such services may obtain the following information from the software to enable you to receive, accept and use certain promotional offers:
 - certain device information, such as your Internet protocol address, the type of operating system and browser you are using, and the name and version of the software you are using,
 - the requested content, and
 - the language code of the device where you installed the software.
 - Your use of the Media Center features to connect to those services serves as your consent to the collection and use of such information.
 - Media Playback Updates. The software on the device may include media playback features which receives updates directly from the MSCORP Media Playback Update servers. If activated by your manufacturer, these updates will be downloaded and installed without further notice to you. The manufacturer is responsible for ensuring these updates work on your device.
 - Windows Update Agent. The software on the device includes Windows Update Agent (WUA). This feature enables your device to access Windows Updates either directly from MSCORP Windows Update server or from a server installed with the required server component and from the Microsoft Windows Update server. To enable the proper functioning of the Windows Update service in the software (if you use it) updates or downloads to the Windows Update service will be required from time to time and downloaded and installed without further notice to you. Without limiting any other disclaimer in these license terms or any license terms accompanying a Windows Update, you acknowledge and agree that no warranty is provided by Microsoft Corporation or their affiliates with respect to any Windows Update that you install or attempt to install on your device.
 - b. Use of Information. Microsoft may use the device information, error reports, and Malware reports to improve our software and services. We may also share it with others, such as hardware and software vendors. They may use the information to improve how their products run with Microsoft software.
 - c. Misuse of Internet-based Services. You may not use these services in any way that could harm them or impair anyone else's use of them. You may not use the services to try to gain unauthorized access to any service, data, account or network by any means.
7. **PRODUCT SUPPORT** . Contact Agilent Technologies for support options. Refer to the support number provided with the device.
8. **MICROSOFT .NET BENCHMARK TESTING**. The software includes one or more components of the .NET Framework (.NET Components). You may conduct internal benchmark testing of those components. You may

disclose the results of any benchmark test of those components, provided that you comply with the conditions set forth at go.microsoft.com/fwlink/?LinkID=66406.

Notwithstanding any other agreement you may have with Microsoft, if you disclose such benchmark test results, Microsoft shall have the right to disclose the results of benchmark tests it conducts of your products that compete with the applicable .NET Component, provided it complies with the same conditions set forth at go.microsoft.com/fwlink/?LinkID=66406.

9. **BACKUP COPY.** You may make one backup copy of the software. You may use it only to reinstall the software on the device.
10. **DOCUMENTATION.** Any person that has valid access to your device or internal network may copy and use the documentation for your internal, reference purposes.
11. **PROOF OF LICENSE.** If you acquired the software on the device, or on a disc or other media, a genuine Certificate of Authenticity label with a genuine copy of the software identifies licensed software. To be valid, this label must be affixed to the device, or included on or in Agilent Technologies software packaging. If you receive the label separately, it is not valid. You should keep the label on the device or packaging to prove that you are licensed to use the software. To identify genuine Microsoft software, see www.howtotell.com.
12. **TRANSFER TO A THIRD PARTY.** You may transfer the software only with the device, the Certificate of Authenticity label, and these license terms directly to a third party. Before the transfer, that party must agree that these license terms apply to the transfer and use of the software. You may not retain any copies of the software including the backup copy.
13. **NOTICE ABOUT THE H.264/AVC VISUAL STANDARD, THE VC-1 VIDEO STANDARD, THE MPEG-4 VISUAL STANDARD AND THE MPEG-2 VIDEO STANDARD.** This software may include H.264/AVC, VC-1, MPEG-4 Part 2, and MPEG-2 visual compression technology. If the software includes those visual compression technologies MPEG LA, L.L.C. requires this notice:

THIS PRODUCT IS LICENSED UNDER ONE OR MORE VIDEO PATENT PORTFOLIO LICENSES SUCH AS, AND WITHOUT LIMITATION, THE AVC, THE VC-1, THE MPEG-4 PART 2 VISUAL, AND THE MPEG 2 VIDEO PATENT PORTFOLIO LICENSES FOR THE PERSONAL AND NON-COMMERCIAL USE OF A CONSUMER TO (i) ENCODE VIDEO IN COMPLIANCE WITH THE ABOVE STANDARDS (VIDEO STANDARDS) AND/OR (ii) DECODE VIDEO THAT WAS ENCODED BY A CONSUMER ENGAGED IN A PERSONAL AND NON-COMMERCIAL ACTIVITY OR WAS OBTAINED FROM A VIDEO PROVIDER LICENSED TO PROVIDE VIDEO UNDER SUCH PATENT PORTFOLIO LICENSES. NONE OF THE LICENSES EXTEND TO ANY OTHER PRODUCT REGARDLESS OF WHETHER SUCH PRODUCT IS INCLUDED WITH THIS PRODUCT IN A SINGLE ARTICLE. NO LICENSE IS GRANTED OR SHALL BE IMPLIED FOR ANY OTHER USE. ADDITIONAL INFORMATION MAY BE OBTAINED FROM MPEG LA, L.L.C. SEE WWW.MPEGLA.COM.
14. **NOTICE ABOUT THE MP3 AUDIO STANDARD.** This software includes MP3 audio encoding and decoding technology as defined by ISO/IEC 11172-3 and ISO/IEC 13818-3. It is not licensed for any implementation or distribution in any commercial product or service.
15. **NOT FAULT TOLERANT.** The software is not fault tolerant. Agilent Technologies installed the software on the device and is responsible for how it operates on the device.
16. **RESTRICTED USE.** The Microsoft software was designed for systems that do not require fail-safe performance. You may not use the Microsoft software in any device or system in which a malfunction of the software would result in foreseeable risk of injury or death to any person. This includes operation of nuclear facilities, aircraft navigation or communication systems and air traffic control.
17. **NO WARRANTIES FOR THE SOFTWARE.** The software is provided as is. You bear all risks of using it. Microsoft gives no express warranties, guarantees or conditions. Any warranties you receive regarding the device or the software do not originate from, and are not binding on, Microsoft or its affiliates. When allowed by your local laws, Agilent Technologies and Microsoft exclude implied warranties of merchantability, fitness for a particular purpose and non-infringement.

18. LIABILITY LIMITATIONS. You can recover from Microsoft and its affiliates only direct damages up to two hundred fifty U.S. Dollars (U.S. \$250.00). You cannot recover any other damages, including consequential, lost profits, special, indirect or incidental damages.

This limitation applies to:

- " anything related to the software, services, content (including code) on third party internet sites, or third party programs, and
- " claims for breach of contract, breach of warranty, guarantee or condition, strict liability, negligence, or other tort to the extent permitted by applicable law.

It also applies even if Microsoft should have been aware of the possibility of the damages. The above limitation may not apply to you because your country may not allow the exclusion or limitation of incidental, consequential or other damages.

19. EXPORT RESTRICTIONS. The software is subject to United States export laws and regulations. You must comply with all domestic and international export laws and regulations that apply to the software. These laws include restrictions on destinations, end users and end use. For additional information, see www.microsoft.com/exporting.

20. ENTIRE AGREEMENT. This agreement, additional terms (including any printed-paper license terms that accompany the software and may modify or replace some or all of these terms), and the terms for supplements, updates, Internet-based services and support services that you use, are the entire agreement for the software and support services.

21. APPLICABLE LAW

a. United States. If you acquired the software in the United States, Washington state law governs the interpretation of this agreement and applies to claims for breach of it, regardless of conflict of laws principles. The laws of the state where you live govern all other claims, including claims under state consumer protection laws, unfair competition laws, and in tort.

b. Outside the United States. If you acquired the software in any other country, the laws of that country apply.

22. Third Party Programs. Microsoft provides the following copyright notices for third party software included in the software. These notices are required by the respective copyright holders and do not change your license to use this software.

Portions of this software are based in part on the work of Spider Systems ® Limited. Because Microsoft has included the Spider Systems Limited software in this product, Microsoft is required to include the following text that accompanied such software:

Copyright 1987 Spider Systems Limited

Copyright 1988 Spider Systems Limited

Copyright 1990 Spider Systems Limited

Portions of this software are based in part on the work of Seagate Software.

Portions of this software are based in part on the work of ACE*COMM Corp. Because Microsoft has included the ACE*COMM Corp. software in this product, Microsoft is required to include the following text that accompanied such

software:

Copyright 1995-1997 ACE*COMM Corp

Portions of this software are based in part on the work of Sam Leffler and Silicon Graphics, Inc. Because Microsoft has included the Sam Leffler and Silicon Graphics software in this product, Microsoft is required to include the following text that accompanied such software:

Copyright © 1988-1997 Sam Leffler

Copyright © 1991-1997 Silicon Graphics, Inc.

Permission to use, copy, modify, distribute, and sell this software and its documentation for any purpose is hereby granted without fee, provided that (i) the above copyright notices and this permission notice appear in all copies of the software and related documentation, and (ii) the names of Sam Leffler and Silicon Graphics may not be used in any advertising or publicity relating to the software without the specific, prior written permission of Sam Leffler and Silicon Graphics.

THE SOFTWARE IS PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EXPRESS, IMPLIED OR OTHERWISE, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

IN NO EVENT SHALL SAM LEFFLER OR SILICON GRAPHICS BE LIABLE FOR ANY SPECIAL, INCIDENTAL, INDIRECT OR CONSEQUENTIAL DAMAGES OF ANY KIND, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER OR NOT ADVISED OF THE POSSIBILITY OF DAMAGE, AND ON ANY THEORY OF LIABILITY, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Portions Copyright © 1998 PictureTel Corporation

Portions of this software are based in part on the work of Highground Systems. Because Microsoft has included the Highground Systems software in this product, Microsoft is required to include the following text that accompanied such software:

Copyright © 1996-1999 Highground Systems

Windows 7 incorporates compression code from the Info-ZIP group. There are no extra charges or costs due to the use of this code, and the original compression sources are freely available from www.info-zip.org/ or <ftp://ftp.info-zip.org/pub/infozip/src/> on the Internet.

Portions Copyright © 2000 SRS Labs, Inc

This product includes software from the 'zlib' general purpose compression library.

Portions of this software are based in part on the work of ScanSoft, Inc. Because Microsoft has included the ScanSoft, Inc. software in this product, Microsoft is required to include the following text that accompanied such software:

TextBridge® OCR © by ScanSoft, Inc.

Portions of this software are based in part on the work of University of Southern California. Because Microsoft has included the University of Southern California software in this product, Microsoft is required to include the following text that accompanied such software:

Copyright © 1996 by the University of Southern California

All rights reserved.

Permission to use, copy, modify, and distribute this software and its documentation in source and binary forms for any purpose and without fee is hereby granted, provided that both the above copyright notice and this permission notice appear in all copies, and that any documentation, advertising materials, and other materials related to such distribution and use acknowledge that the software was developed in part by the University of Southern California, Information Sciences Institute. The name of the University may not be used to endorse or promote products derived from this software without specific prior written permission.

THE UNIVERSITY OF SOUTHERN CALIFORNIA makes no representations about the suitability of this software for any purpose. THIS SOFTWARE IS PROVIDED "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Other copyrights might apply to parts of this software and are so noted when applicable.

Portions of this software are based in part on the work of James Kanze. Because Microsoft has included the James Kanze software in this product, Microsoft is required to include the following text that accompanied such software:

COPYRIGHT AND PERMISSION NOTICE

All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, publish, distribute, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, provided that the above copyright notice(s) and this permission notice appear in all copies of the Software and that both the above copyright notice(s) and this permission notice appear in supporting documentation. Permission is also given to modify the software to any extent, under the condition that, in the modified software, the prefix "GB_" is changed to something else, and the name directories for includes files ("gb" in this distribution) is also changed.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE BE LIABLE FOR ANY CLAIM, OR

ANY SPECIAL INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

This product contains software from Cisco ISAKMP Services.

Portions of this software are based in part on the work of RSA Data Security, Inc. Because Microsoft has included the RSA Data Security, Inc. software in this product, Microsoft is required to include the following text that accompanied such software:

Copyright © 1990, RSA Data Security, Inc. All rights reserved.

License to copy and use this software is granted provided that it is identified as the "RSA Data Security, Inc. MD5 Message-Digest Algorithm" in all material mentioning or referencing this software or this function. License is also granted to make and use derivative works provided that such works are identified as "derived from the RSA Data Security, Inc. MD5 Message-Digest Algorithm" in all material mentioning or referencing the derived work.

RSA Data Security, Inc. makes no representations concerning either the merchantability of this software or the suitability of this software for any particular purpose. It is provided "as is" without express or implied warranty of any kind.

These notices must be retained in any copies of any part of this documentation and/or software.

Portions of this software are based in part on the work of OpenVision Technologies, Inc. Because Microsoft has included the OpenVision Technologies, Inc. software in this product, Microsoft is required to include the following text that accompanied such software:

Copyright 1993 by OpenVision Technologies, Inc.

Permission to use, copy, modify, distribute, and sell this software and its documentation for any purpose is hereby granted without fee, provided that the above copyright notice appears in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of OpenVision not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. OpenVision makes no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

OPENVISION DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL OPENVISION BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Portions of this software are based in part on the work of Regents of The University of Michigan. Because Microsoft has included the Regents of The University of Michigan software in this product, Microsoft is required to include the following text that accompanied such software:

Copyright © 1995, 1996 Regents of The University of Michigan.

All Rights Reserved.

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appears in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of The University of Michigan not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. This software is supplied as is without expressed or implied warranties of any kind.

Copyright © 1993, 1994 Regents of the University of Michigan.

All rights reserved.

Redistribution and use in source and binary forms are permitted provided that this notice is preserved and that due credit is given to the University of Michigan at Ann Arbor. The name of the University may not be used to endorse or promote products derived from this software without specific prior written permission. This software is provided "as is" without express or implied warranty.

Portions of this software are based in part on the work of Massachusetts Institute of Technology. Because Microsoft has included the Massachusetts Institute of Technology software in this product, Microsoft is required to include the following text that accompanied such software:

Copyright 1989, 1990 by the Massachusetts Institute of Technology. All Rights Reserved.

Export of this software from the United States of America may require a specific license from the United States Government. It is the responsibility of any person or organization contemplating export to obtain such a license before exporting.

WITHIN THAT CONSTRAINT, permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and

this permission notice appear in supporting documentation, and that the name of M.I.T. not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. M.I.T. makes no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

Under U.S. law, this software may not be exported outside the US without license from the U.S. Commerce department.

Copyright 1994 by the Massachusetts Institute of Technology. All Rights Reserved.

Export of this software from the United States of America may require a specific license from the United States Government. It is the responsibility of any person or organization contemplating export to obtain such a license before exporting.

WITHIN THAT CONSTRAINT, permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of M.I.T. not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. M.I.T. makes no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

This product includes software developed by the University of California, Berkeley and its contributors.

Portions of this software are based in part on the work of the "Entrust" security technology licensed from Northern Telecom.

Portions of this software are based in part on the work of Hewlett-Packard Company. Because Microsoft has included the Hewlett-Packard Company software in this product, Microsoft is required to include the following text that accompanied such software:

Copyright © 1994 Hewlett-Packard Company

Permission to use, copy, modify, distribute and sell this software and its documentation for any purpose is hereby granted without fee, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation. Hewlett-Packard Company and Microsoft Corporation make no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

This product includes software from the 'libpng' PNG reference library.

Portions of this software are based in part on the work of Autodesk, Inc. Because Microsoft has included the Autodesk, Inc. software in this product, Microsoft is required to include the following text that accompanied such software:

© Copyright 1995 by Autodesk, Inc.

This product contains graphics filter software; this software is based in part on the work of the Independent JPEG Group.

This product includes True Verb technology from KS Waves Ltd.

Portions of this software are based in part on the work of SGS-Thomson Microelectronics, Inc. Because Microsoft has included the SGS-Thomson Microelectronics, Inc. software in this product, Microsoft is required to include the following text that accompanied such software:

Copyright 1996 SGS-Thomson Microelectronics, Inc. All Rights Reserved

Portions of this software are based in part on the work of Unicode, Inc. Because Microsoft has included the Unicode, Inc. software in this product, Microsoft is required to include the following text that accompanied such software:

COPYRIGHT AND PERMISSION NOTICE

Copyright © 1991-2005 Unicode, Inc. All rights reserved. Distributed under the Terms of Use in www.unicode.org/copyright.html.

Permission is hereby granted, free of charge, to any person obtaining a copy of the Unicode data files and any associated documentation (the "Data Files") or Unicode software and any associated documentation (the "Software") to deal in the Data Files or Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, and/or sell copies of the Data Files or Software, and to permit persons to whom the Data Files or Software are furnished to do so, provided that (a) the above copyright notice(s) and this permission notice appear with all copies of the Data Files or Software, (b) both the above copyright notice(s) and this permission notice appear in associated documentation, and © there is clear notice in each modified Data File or in the Software as well as in the documentation associated with the Data File(s) or Software that the data or software has been modified.

THE DATA FILES AND SOFTWARE ARE PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE BE LIABLE FOR ANY CLAIM, OR ANY SPECIAL INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THE DATA FILES OR SOFTWARE.

Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in these Data Files or Software without prior written authorization of the copyright holder.

The Combined PostScript Driver was the result of a cooperative development process by Adobe Systems Incorporated and Microsoft Corporation.

Portions of this software are based in part on the work of Media Cybernetics. Because Microsoft has included the Media Cybernetics software in this product, Microsoft is required to include the following text that accompanied such software:

HALO Image File Format Library © 1991-1992 Media Cybernetics, Inc.

Portions of this software are based in part on the work of Luigi Rizzo. Because Microsoft has included the Luigi Rizzo software in this product, Microsoft is required to include the following text that accompanied such software:

© 1997-98 Luigi Rizzo (luigi@iet.unipi.it)

Portions derived from code by Phil Karn (karn@ka9q.ampr.org), Robert Morelos-Zaragoza (robert@spectra.eng.hawaii.edu) and Hari Thirumorthy (harit@spectra.eng.hawaii.edu), Aug 1995

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHORS AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT

LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Portions of this software are based in part on the work of W3C. Because Microsoft has included the W3C software in this product, Microsoft is required to include the following text that accompanied such software:

W3C ® SOFTWARE NOTICE AND LICENSE

www.w3.org/Consortium/Legal/2002/copyright-software-20021231

This work (and included software, documentation such as READMEs, or other related items) is being provided by the copyright holders under the following license. By obtaining, using and/or copying this work, you (the licensee) agree that you have read, understood, and will comply with the following terms and conditions.

Permission to copy, modify, and distribute this software and its documentation, with or without modification, for any purpose and without fee or royalty is hereby granted, provided that you include the following on ALL copies of the software and documentation or portions thereof, including modifications:

1. The full text of this NOTICE in a location viewable to users of the redistributed or derivative work.
2. Any pre-existing intellectual property disclaimers, notices, or terms and conditions. If none exist, the W3C Software Short Notice should be included (hypertext is preferred, text is permitted) within the body of any redistributed or derivative code.
3. Notice of any changes or modifications to the files, including the date changes were made. (We recommend you provide URLs to the location from which the code is derived.)

THIS SOFTWARE AND DOCUMENTATION IS PROVIDED "AS IS," AND COPYRIGHT HOLDERS MAKE NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE OR THAT THE USE OF THE SOFTWARE OR DOCUMENTATION WILL NOT INFRINGE ANY THIRD PARTY PATENTS, COPYRIGHTS, TRADEMARKS OR OTHER RIGHTS.

COPYRIGHT HOLDERS WILL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF ANY USE OF THE SOFTWARE OR DOCUMENTATION.

The name and trademarks of copyright holders may NOT be used in advertising or publicity pertaining to the software without specific, written prior permission. Title to copyright in this software and any associated

documentation will at all times remain with copyright holders.

Portions of this software are based in part on the work of Sun Microsystems, Inc. Because Microsoft has included the Sun Microsystems, Inc. software in this product, Microsoft is required to include the following text that accompanied such software:

Sun RPC is a product of Sun Microsystems, Inc. and is provided for unrestricted use provided that this legend is included on all tape media and as a part of the software program in whole or part. Users may copy or modify Sun RPC without charge, but are not authorized to license or distribute it to anyone else except as part of a product or program developed by the user.

SUN RPC IS PROVIDED AS IS WITH NO WARRANTIES OF ANY KIND INCLUDING THE WARRANTIES OF DESIGN, MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, OR ARISING FROM A COURSE OF DEALING, USAGE OR TRADE PRACTICE.

Sun RPC is provided with no support and without any obligation on the part of Sun Microsystems, Inc. to assist in its use, correction, modification or enhancement.

SUN MICROSYSTEMS, INC. SHALL HAVE NO LIABILITY WITH RESPECT TO THE INFRINGEMENT OF COPYRIGHTS, TRADE SECRETS OR ANY PATENTS BY SUN RPC OR ANY PART THEREOF.

In no event will Sun Microsystems, Inc. be liable for any lost revenue or profits or other special, indirect and consequential damages, even if Sun has been advised of the possibility of such damages.

Sun Microsystems, Inc.

2550 Garcia Avenue

Mountain View, California 94043

Manufactured under license from Dolby Laboratories. Dolby and the double-D symbol are trademarks of Dolby Laboratories. Confidential unpublished works. Copyright 1992-1997 Dolby Laboratories. All rights reserved.

Portions of this software are based in part on the work of Andrei Alexandrescu. Because Microsoft has included the Andrei Alexandrescu software in this product, Microsoft is required to include the following text that accompanied such software:

The Loki Library

Copyright © 2001 by Andrei Alexandrescu

This code accompanies the book:

Alexandrescu, Andrei. "Modern C++ Design: Generic Programming and Design Patterns Applied." Copyright © 2001. Addison-Wesley.

Permission to use, copy, modify, distribute and sell this software for any purpose is hereby granted without fee, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation.

The author or Addison-Wesley Longman make no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

Portions Copyright © 1995 by Jeffrey Richter

Portions of this software are based in part on the work of the Distributed Management Task Force, Inc. (DMTF). Because Microsoft has included software based on DMTF specifications in this product, Microsoft is required to include the following text:

Copyright © 2007 Distributed Management Task Force, Inc. (DMTF). All rights reserved.

Portions of this work are derived from "The Draft Standard C++ Library" Copyright © 1995 by P.J. Plauger published by Prentice-Hall and are used with permission.

Portions of this software are based in part on the work of Hewlett-Packard Company. Because Microsoft has included the

Hewlett-Packard Company software in this product, Microsoft is required to include the following text that accompanied such software:

Copyright © 2002, 2003 Hewlett-Packard Company.

About Notice:

This software is based on software available from mpvtools.sourceforge.net.

This software processes a format called MPV. MPV is an open specification for managing collections and multimedia playlists of photo, video, and music content and associated metadata and is available at no cost from the Optical Storage Technology Association. More information about the MPV specification can be found at www.osta.org/mpv.

Permission Notice:

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice, this permission notice, and the above About Notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR

OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

All other trademarks are property of their respective owners.

PNA User Accounts and Passwords

Important: When the PNA power is switched on, it AUTOMATICALLY logs into Windows using the default user name and password. You do NOT need to log on. This gives anyone full access to the analyzer. The following steps can be taken to increase security of your PNA.

- Require users to logon when the PNA computer is turned ON - [Learn how to enable this feature](#)
- Setup individual accounts on the PNA with varying level of access - Learn how to [Add or Change User Accounts and Passwords](#)

[Please read about Anti-virus protection for your PNA](#)

Existing User Accounts

The following user accounts already exist on new PNAs.

- **Default User Account** The Default User Account cannot be deleted. We recommend you change the password and, if desired, the user name. DO NOT FORGET YOUR NEW PASSWORD. If you choose to [require users to logon when the PNA is turned ON](#), you will not be able to start your PNA without it.
 - User name (not case sensitive): **PNA-USER**
 - Password (case sensitive): **pna**
- **PNA-SERVICE** This limited user account is created by Agilent for service purposes. Each PNA has a unique password for this account. Although allowed by Windows, please do not delete this account.

Notes

- You can create as many user accounts as you like.
- The user name is not case sensitive. The password IS case sensitive.
- The PNA local policies are set so that, if logon is required, you must retype the user name (and password) every time. Do not change the local policies on the PNA.

Add or Change User Accounts, Passwords, and require Logon

If the analyzer is in a secure environment, you can setup PNA users by name and grant various levels of access. This is particularly important when the PNA is remotely controlled or accessed over LAN.

You can designate a person as the administrator and then configure the PNA to allow others to use it with reduced permissions. That is, other people can be signed on to use the analyzer but they will not

have the ability to perform all of the administrative functions that you can as the administrator.

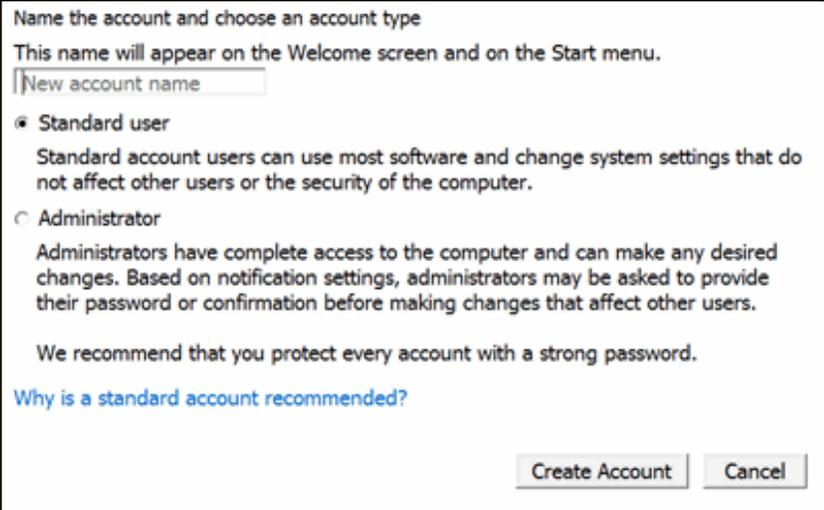
The following are examples of some of the functions that can be performed with these account types:

- **Administrator** - Can download and install drivers. Can save and delete all files.
- **Standard User** - Can save and delete state and data files. Can NOT save PNA [service adjustment](#) files.

Note: To connect and install a new ECal module, you must be logged on with an Administrator account. A user account with limited access will NOT allow an ECal module driver to be loaded. [Learn more.](#)

Learn more about [Microsoft User Account Permissions](#) (internet connection required).

How to add a user account and require logon

Windows 7
Click Start , then Control Panel
Click User Accounts
Click Manage another account.
CAUTION: Although allowed by Windows, do NOT allow an Administrator account without a password. Internet viruses look for, and exploit, this condition.
Click Create a new account.
Enter an account name.
Select an account type.
Click Create Account.


Note:

A standard user account will NOT allow an ECal module driver to be loaded. [Learn more.](#)

However, standard access users **ARE** able to install firmware.

Optional: Click **Create a password** to require the user to enter a password when logging on.

Last Modified:

25-Nov-2013	Updated for windows 7
14-Sep-2010	Updated default account
8-Apr-2008	Added Logon notes

PNA Computer Properties

The PNA uses a personal computer and a Windows operating system. The following are common tasks that you may need to perform on the PNA computer.

- [View or change Full Computer Name](#)
- [Check IP Address](#)
- [Check the amount of RAM](#)
- [Check CPU Speed](#)
- [Check HDD Capacity](#)
- [Set Time and Date](#)
- [Internal and External Speakers](#)

Other Administrative Task Topics

View or change Full Computer Name

Your PNA has a unique computer name that identifies it on a network. To view or change the computer name, you must first [minimize the PNA](#) application.

Windows 7
On the desktop, right-click My computer Icon
Click Properties
Scroll down to Computer name, domain, and workgroup settings .
To make changes, click Change Settings .

Note: To add your computer to a domain, or to set up the networking configuration, contact your company's I.T. department. This setup is custom for each company.

To restore the PNA application, click **PNA Analyzer** in the task bar at the bottom of the screen.

Check IP Address

If your PNA is connected to a LAN, you can view the IP address and other networking information.

1. [Minimize the PNA](#) application
2. Click **Start**
3. In the Search field, type **cmd**, then click **OK**
4. At a DOS prompt, type **ipconfig /all**

Check the amount of RAM

Random Access Memory (RAM) is the amount of working memory in your computer.

The amount of RAM in your PNA may limit your ability to upgrade firmware. See http://na.tm.agilent.com/pna/firmware/PNA_support_matrix.doc

To view the amount of PNA RAM, you must first [minimize the PNA](#) application.

Windows 7
On the desktop, right-click My computer Icon
Click Properties
The amount of RAM appears under System

To restore the PNA application, click **PNA Analyzer** in the task bar at the bottom of the screen.

Check CPU Speed

The speed of the PNA processor (CPU) is a factor in determining how quickly the PNA processes data. Also, the CPU speed in your PNA may limit your ability to upgrade firmware. See http://na.tm.agilent.com/pna/firmware/PNA_support_matrix.doc.

You can see which CPU is in your PNA by comparing your PNA rear-panel with the images at <http://na.tm.agilent.com/pna/cputype.html>.

Or, beginning with PNA Rev. A.09.10, on the PNA click **Help**, then **About Network Analyzer**. [Learn more.](#)

Or, you can do the following to check your PNA CPU speed:

1. [Minimize the PNA](#) application.
2. Then do the following:

Windows 7
On the desktop, right-click My computer Icon
Click Properties
The CPU speed appears under System

To restore the PNA application, click **PNA Analyzer** in the task bar at the bottom of the screen.

Check Hard Disk Drive (HDD) or Solid State Drive (SSD) Capacity

To view the capacity of drive space, you must first [minimize the PNA](#) application.

Then do the following:

Windows 7
On the desktop, right-click the Start button
Then Open Windows Explorer
In the left pane, click Computer
The amount of free drive space and capacity for each of the partitions appears.

[Learn how to determine your HDD part number](#) (Internet connection required).

Set Time and Date

To set the time and date on your PNA, you must first [minimize the PNA](#) application.

1. Move the cursor to the lower corner of the screen
2. When the taskbar appears, double-click on the displayed time. This opens the **Date/Time Properties** dialog box.
3. Change the date, time, and time zone as appropriate.

To restore the PNA application, click **PNA Analyzer** in the task bar at the bottom of the screen.

Internal Speaker

There is an internal sound card and speaker in the PNA. However, there is no audio output jack.

There may be times when you might want to control the speaker volume or turn the speaker OFF, such as when the PNA is generating errors. [Learn more about errors.](#)

To control the PNA speaker volume:

1. Press **System**, then **Service**, then **Utilities**, then **Speaker Volume**
 2. Enter a value between **0** (speaker OFF) and **100** (highest volume)
-

Last Modified:

5-Apr-2012	Remove Win 2000
17-Jun-2011	Modified speaker volume
30-Nov-2009	Added Help, About note.
26-Nov-2008	Added external speaker note
12-Mar-2008	Added link to support site
20-Sep-2007	Added speaker OFF

Run Error Check and Disk Defragmenter

When the PNA is shutdown unexpectedly or power is removed without first shutting down, large amounts of Hard Disk Drive space is rendered unusable. If shutdown in this manner enough times, the PNA could become unstable and no longer work.

This Hard Disk Drive space can be recovered by first running Windows **Error-checking** to find and correct errors on the disk, and then the **Disk Defragmenter** to recover Hard Disk Drive space.. These programs should be run routinely, about every 1 to 4 weeks, depending on how often the PNA is unexpectedly shutdown.

To learn more about Disk Defragmenter, see the Windows Help file.

Follow this procedure to run these programs:

Windows 7
On the desktop, double-click My Computer
In the upper pane, click Properties
Click the Tools tab

Error-checking

- Click **Check Now**.
- Check **Automatically fix file system errors**.
- Click **Start**.
- Click **Yes** to run disk check on next restart.
- Manually [restart the PNA](#). The disk check will run before Windows restarts.

Approximately every six months, check the second box in addition to the first box. The error-checking process takes much longer, but performs a more complete check.

Defragmentation

- Click **Defragment Now...**
- Click **Defragment** to begin the defragment process.
- Click **Close** when defragmentation is complete.

Last modified:

25-Jun-2013 Modified for Windows 7

Recovering from PNA Hard Drive Problems

The leading cause of PNA failures is problems with the PNA Hard Disk Drive (HDD). These problems are usually preventable (see [Preventing PNA HDD Problems](#)), and in many cases, recoverable. The following could save you weeks of downtime and the cost of replacing your PNA HDD.

This document is now on the Agilent PNA Support Website: <http://na.tm.agilent.com/pna/>. When at this webpage, click the **Hard Drive Recovery** link.

If your PNA does experience a Hard Disk Drive Problem, you will not be able to access this Help file, but you may be able to access the Internet from another computer.

Wake PNA at Specified Time

Depending on the CPU that is used in your PNA-X, you may be able to start your PNA-X at a specified time every day. This feature allows the PNA-X to be shut down at night (saving energy), then wake the PNA-X at a specified time to allow sufficient warm-up time in the morning before being used.

To use this feature, the PNA-X must have a 2000 MHz CPU board. [See how to determine the CPU board.](#)

How to configure the PNA-X to wake at a specified time:

1. Connect a USB keyboard to the PNA-X.
2. [Restart](#) the PNA-X.
3. During power-up, press F2 on the keyboard at the moment it indicates to do so on the bottom of the PNA-X screen.
4. In the BIOS, go to **Configuration** then **Power Control Configuration**. Set **Resume On Time:** to **ON**, and set the **Resume Time:** to the desired wake-up time.
5. Press **F10** to Save, Exit, and restart the PNA-X.
6. At the end of the day, Shut Down (not hibernate or standby) the PNA-X.
7. The PNA-X will wake-up at the specified time.

Wake ON LAN (WOL)

Note: A utility can also be set to provide a WOL call. This is outside the scope of PNA Help. Search the internet for **WOL utility**.

To configure the PNA for Wake ON LAN (WOL)

Follow the above procedure for accessing the BIOS.

Then, in the BIOS:

For newer i7 and Celeron CPUs:

1. Go to **Chipset** menu, then **South Bridge**, then **SB PCH Options**.
2. Under the PCH LAN Controller, set WOL to **Enabled**.
3. Press **F10** to Save, Exit.

For all older CPUs:

1. Go to **Configuration** then **LAN Configuration**. Ensure that **Power Saving when S5** to **Disabled** (default setting).
2. Go to **Configuration** then **Power Control Configuration**. Set **PME Wake From S5** to **Enabled**.
3. Press **F10** to Save, Exit.

To must also do the following on the PNA:

1. From the PNA desktop, right-click **My Computer**, then **Management**.
 2. In the left pane, under **System Tools**, click **Device Manager**.
 3. in the right pane, click **+** in front of **Network Adapters**.
 4. Under **Network Adapters**, right click **Intel (R) 82577LRM Network Connection**, then **Properties**.
 5. On the **Power Management** tab, check to enable **Magic Packet from power off state**.
-

Last Modified:

1-Mar-2013	Added WOL for newer CPUs
18-May-2010	MX New topic

Microsoft Windows Considerations

In this topic:

- [Microsoft Windows on the PNA](#)
- [LAN Connections](#)
- [Single and Double Click option](#)
- [Windows XP Theme](#)
- [Printing](#)

See Also

[Windows 7 File Locations](#)

[Your Programs on Windows 7](#)

[Microsoft EULA](#)

Microsoft Windows on the PNA

Beginning in Sept. 2013, the PNA is shipped from the factory with **Windows[®] Embedded Standard 7** operating system. This OS supports both 32-bit and 64-bit applications. Therefore, the file structure has changed slightly. ONLY PNA models with Windows 7 can upgrade to FW A.10.00 and above.

[Learn about Windows 7 file locations on the PNA.](#)

VERY IMPORTANT Protect your hard drive!

The leading cause of PNA failures is problems with the PNA Hard Disk Drive (HDD). These problems are usually preventable, and in many cases, recoverable. [Learn more about protecting your PNA.](#)

Using USB

The PNA has USB ports on the front panel and on the rear panel. The main advantages of USB are “hot” connects and disconnects and fast data transfer speeds. Electronic Calibration modules are also available with USB connections.

The first time you plug a device into a USB port there is some wait time. Windows reports it is identifying the hardware, then searching for the correct driver, then installing the driver (if it was found).

Connecting that same device back into that same port later is quick and easy, but if you move the device to a different USB port, you will have to wait through the hardware ID and driver search again.

[Learn about USB limitations.](#)

LAN Connections

Windows supports DHCP and fixed IP addressing. Also, “Hot” connect and disconnect of the LAN cable, as well as a visual indicator of LAN status in system tray area, makes LAN connections more intuitive. In addition, the Hardware Wizard helps users with system hardware configuration.

Single and Double Click option

By default, Windows allows a single-click method of launching icons. To revert to double-clicking, click **Start**, then **Settings**, then **Control Panel**, then click **Mouse**. In the Mouse Properties dialog, select **Double-click to open an item**. Then click **OK**.

Windows Themes

The PNA application is designed for, and best viewed in, **Agilent Technologies** theme. To change the theme:

1. [Minimize the PNA application.](#)
2. Right-click on the Desktop, then click **Personalize**
3. Use the scrollbar, then select a Theme.

Printing

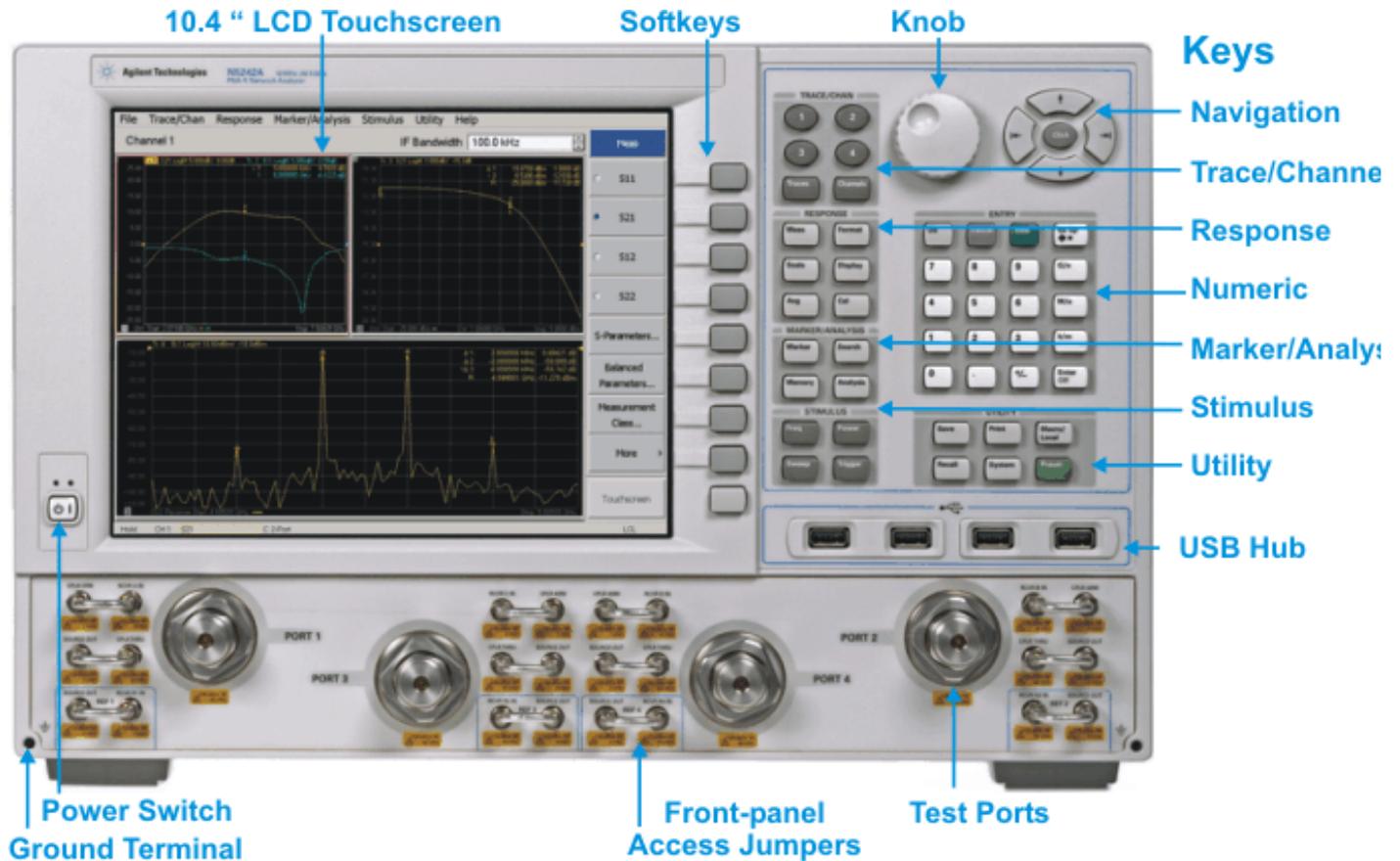
Adding a printer should be done outside of the PNA application. [Learn more.](#)

Last modified:

25-Nov-2013 Modified for Windows 7

PNA-X, N522x, and N523x Front-Panel Tour

Click on an area of the image to learn more.



See Also

- [PNA-X Models/Options](#)
- [N522xA Models/Options](#)
- [N523x Models/Options](#)
- [Display area](#)
- [Rear-panel Tour](#)
 - [PNA-X and N522xA](#)
 - [N523xA](#)

Familiar Hardkey layout, similar to Agilent 8720 and 8753 Network Analyzers

Back to the familiar layout, significantly different from legacy PNA models. Most measurement settings are made from the Stimulus Block and the Response Block.

Fully functional Hardkey/Softkey selections consistent with Menu (mouse) selections

Access ALL PNA settings from the front panel using hardkey/softkeys or from the Menu using a mouse. Both methods are consistent; learn the menu structure once, and it applies to both methods of UI navigation.

Power Switch

Used for choosing between power-on (|) and standby (O) state.

Learn to power ON and OFF the PNA.

Test Ports

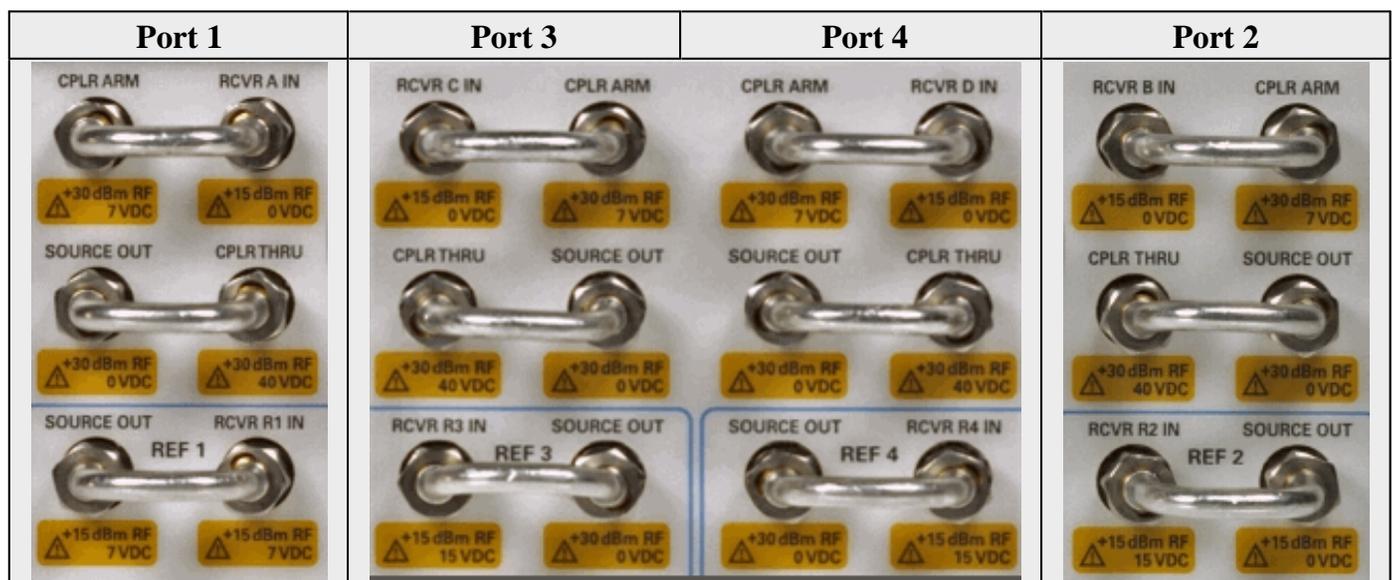
The models are available with 2 or 4 test ports.

See Specs for more information about the Test port connectors and Input damage levels.

Front panel Access Jumpers

These connectors provide direct access to the PNA source and receivers. This allows you to make a wide variety of measurements and improve dynamic range.

See front panel jumpers specifications



N5247A



Port 1 and Port 3 **SW SRC OUT - COMB IN** jumpers moved from rear-panel (J8 through J11) to front-panel to minimize path loss.

Important: All of the N5247A front-panel jumpers are **EXTREMELY FRAGILE** . Learn more .

USB Hub

This USB hub contains four USB ports to power your PNA peripherals. There are also four USB ports on the rear panel.

Limitation: The total power consumption for all eight USB ports is limited to 4.0 amps. If this limit is exceeded, all USB ports are disabled until a device is removed and power consumption falls below the limit. When first connected, Agilent ECal modules 8509x and N4431 draw significantly more current than other modules.

Note:

The **FIRST TIME** each USB device (ECal module, power sensor, and so forth) by serial number is connected to a specific PNA USB port, you must be logged in to the PNA with an **Administrator** account. Learn how .

When a **New Hardware Found** dialog appears, click **OK** to install the device.

After being installed, when that same USB device is connected to that same USB port, you can be logged in to the PNA with a Limited/User account.

Ground terminal

Connect a banana-type plug to this terminal for grounding to the PNA chassis.

No probe power

Probe power is **NOT** provided on PNA models. Learn more about Active Probing

Hardkeys

TRACE/CHAN Keys

Manages the Traces and Channels on the PNA display.

Hard Key	Invokes these Softkeys
1, 2, 3, 4	Makes the corresponding trace active.
Traces	Invokes the Traces softkey menu which allows you to create a new trace, select a trace, delete a trace, or maximize the trace.
Channels	Invokes the Channels softkey menu which allows you to manage channels.

RESPONSE Keys

Performs operations on measurement traces after data is measured - not including Data Analysis operations.

Hard Key	Invokes these Softkeys - Click to learn more
Meas	Measurement selections <ul style="list-style-type: none"> • S-Parameters • Balanced Parameters • Measurement Class More Meas <ul style="list-style-type: none"> • Receivers
Format	Format
Scale	Scale Electrical Delay Phase Offset More <ul style="list-style-type: none"> • Velocity Factor • Media -Waveguide/coax • Waveguide cutoff freq
Display	Display settings Arrangements (Overlay...) Windows (Managing) Measurement Setups

Display Items

- Title
- Trace Status
- Freq Stimulus
- Marker Display
- Toolbars
- Tables
- Status Bar
- Hide Sofkeys
- Minimize App

Avg

Averaging

Smoothing

IF Bandwidth

Cal

Start Cal

- Cal Wizard
- Preferences
- Global Delta Match

Correction

Power Cal

- Source Cal
- Receiver Cal

Manage Cals

- Cal Set
- Cal Type
- Cal Set Viewer

Properties (must have a Cal ON)

Port Ext Toolbar

Interpolation

Fixtures

- ON |Off
- Port matching
- lots more

Manage Cal Kit

Manage ECal

System Z0

Velocity Factor

MARKER/ANALYSIS Keys

Control all aspects of Data Analysis including Markers and Math functions..

Hard Key	Invokes these Softkeys - Click to learn more
Marker	Markers
	Properties <ul style="list-style-type: none">• Delta Markers• Discrete• Type• Coupled
	Functions
Search	Marker Search
Memory	Data/ Memory Math
	8510 Mode
Analysis	Limit Lines

	<ul style="list-style-type: none"> • Limit Test • Global Pass/Fail
	Trace Statistics
	Gating
	Transform
	<ul style="list-style-type: none"> • Windowing • Coupling • Distance Marker
	Equation Editor

STIMULUS Keys

Controls settings that determine **what** data (stimulus range), and **how** data (sweep type and triggering), is measured.

Hard Key	Invokes these Softkeys - Click to learn more
Freq	Frequency Range Frequency Offset Mode
Power	RF Power level Power Slope Power and Attenuator settings
Sweep	Sweep Time Number of Points Sweep Type Sweep Setup Segment Table settings
Trigger	Trigger settings

UTILITY Keys

Performs global PNA operations.

Hard Key	Invokes these Softkeys - Click to learn more
Save	File Save Save (State) As Save Data As Auto Save Manage Files Delete Files User Preset
Print	Print Print to file Page Setup
Macro/Local	Macro Setup Run Macros
Recall	File Recall
System	Security Configure <ul style="list-style-type: none"> • SICL / GPIB • Control Panel (Windows) • System Z0 • Power Meter Settings • Millimeter Module Service Help <ul style="list-style-type: none"> • Error Messages • About NA User Key

	Touchscreen
Preset	Preset
	User Preset

ENTRY Keys

Hard Key	Invokes these Softkeys
OK	Closes a dialog box and enters any values made in the dialog box.
Cancel	Closes a dialog box.
Help	Launches this Help file.
Bk Sp	Back Space. Backs up the cursor and deletes any previous selection.
1 to 9	Selects values for measurement settings, then press Enter or G/n - M/u - k/m to complete the selection.
G/n	Completes the value selection, assigning a unit of measurement.
M/u	<ul style="list-style-type: none"> • G/n (Giga/Nano) E12 or E-12
k/m	<ul style="list-style-type: none"> • M/u (Mega/micro) E6 or E-6 • k/m (kilo/milli) E3 or E-3
Enter	Enters the values that you select for the measurement settings.
Off	
Decimal point	Enters a decimal point to designate fractions of a whole number.
+/-	Plus - Minus Toggles between a positive and negative value entry if it is the first key pressed in the entry.

Knob

Rotate to increase or decrease the value of the active entry.

Navigation Keys

These keys allow you to navigate through menus and dialog boxes and select choices from the active entry toolbar.

Hard Key	Invokes these Softkeys
Left / Right	Moves left and right through menus. Moves tab-left and tab-right within dialog boxes.
Up / Down	Moves up and down through menus. Behaves as follows in a dialog box: <ul style="list-style-type: none"> • Modifies a numeric value • Moves through items in a drop-down list • Moves through options buttons in a group of option buttons
Click	Makes a selection just like a mouse click.

Last Modified:

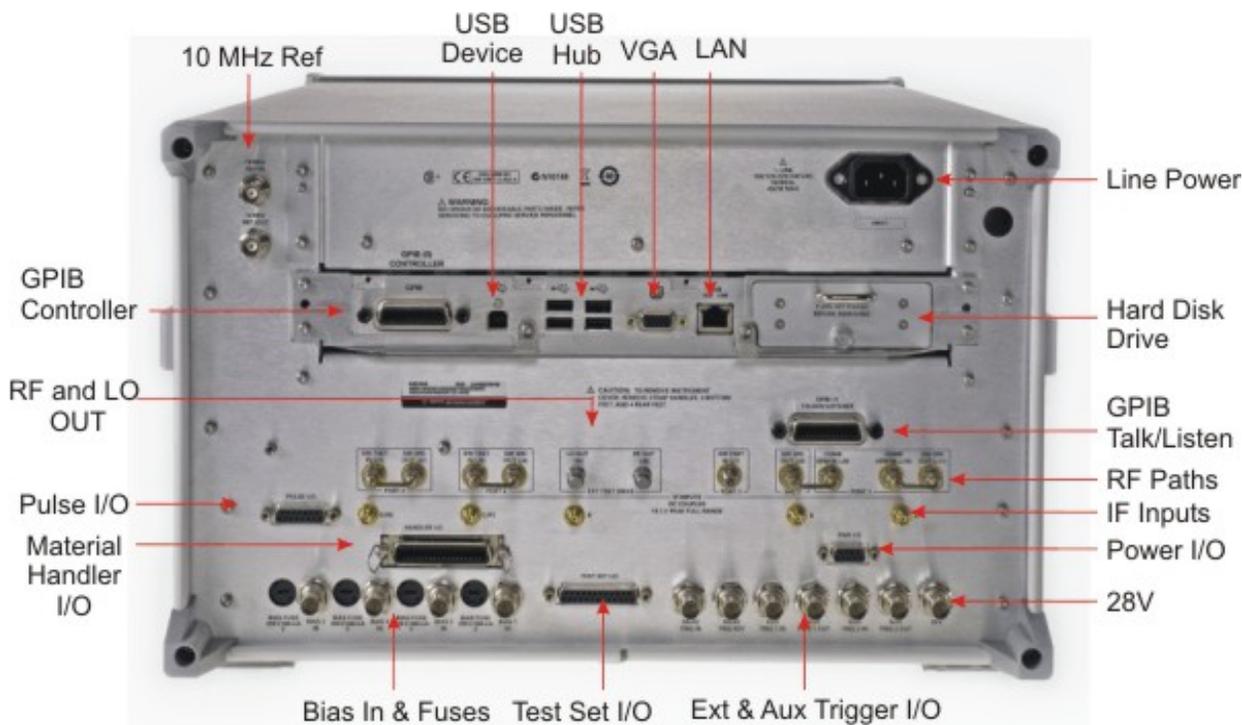
- 10-May-2012 Added N523x models
- 4-May-2011 Added N522xA models
- 15-Apr-2008 Added ALL front panel jumper images
- 23-Aug-2007 Added front panel jumpers image

PNA-X and N522xA Rear Panel

See [Differences between the PNA-X and N522xA Models](#)

See [N523xA rear-panel](#).

Click image to learn more.



10 MHz Reference IN/OUT

10 MHz Reference Input When a 10 MHz external reference signal is detected at this port, it will be used as the instrument frequency reference instead of the internal frequency reference.

10 MHz Reference Output This BNC(f) connector outputs a frequency reference signal for use by other test equipment.

- See [SCPI](#) command that detects an external reference signal at this connector.
- [See Specifications](#)

VGA Connector [Learn more](#)

USB Hub

This USB hub contains four USB ports to power your PNA peripherals. There are also four USB ports on the [front panel](#).

Limitation: The total power consumption for all eight USB ports is limited to 4.0 amps. If this limit is exceeded, all USB ports are disabled until a device is removed and power consumption falls below the limit. When first connected, Agilent ECal modules 8509x and N4431 draw significantly more current than other modules. [See Specifications](#).

[See Important First-time USB connection note.](#)

USB Device [Learn more](#)

LAN Connector

This 10/100BaseT Ethernet connection has a standard 8-pin configuration and auto selects between the two data rates.

Line Power

[See Specifications](#)

GPIB Controller and Talker/Listener Ports

The PNA-X can be a GPIB Controller and Talker/Listener. [Learn more.](#)

RF Path Access

These connectors are NOT available on the [N522x](#) and [N5264A](#) models.

These connectors allow [RF Path Configuration](#).

Ports 3 and 4 are not available on 2-port models.

N5247A - J8 thru J11 are moved to the [front-panel](#).



RF and LO OUT



The **RF OUT** connector is NOT available on the [N5264A](#)

For the [N5247A](#) and [N5227A](#):- Added RF2 OUT (J12) for 4-port 110 GHz single sweep PNA. Enables driving two mmWave modules simultaneously. [Learn more](#).



Caution: LO OUT has more power than previous PNA models.

[See specifications](#)

IF Path Inputs

Option 020 adds these connectors, which allow access to the PNA Receiver / IF paths.



These are labeled A, B, C/R1, D/R2, R.

- For 2-port models, use A, B, R1, R2.
- For 4-port models, use A, B, C, D, R.

[See IF Path Configuration settings and block diagram](#).

Power I/O

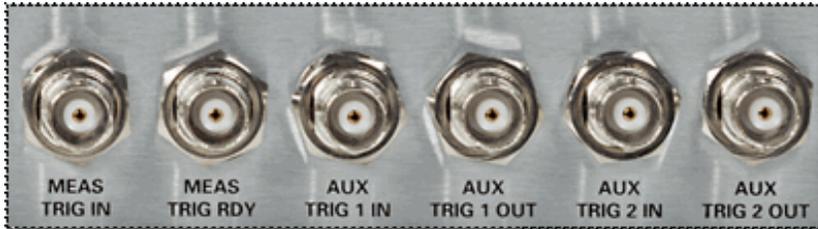
[See Details](#)

28 V (BNC output)



Used to power a noise source for the [Noise Figure App](#).

External and AUX Trigger I/O



MEAS TRIG IN - When enabled, PNA is triggered by signals on this connector. [Learn more.](#)

MEAS TRIG RDY When enabled, PNA outputs a 'READY' signal on this connector to other devices. [Learn more.](#)

AUX TRIG 1&2 IN When enabled, PNA accepts signals on these connectors which indicates that the external devices is ready to be triggered. [Learn more.](#)

AUX TRIG 1&2 OUT When enabled, PNA outputs signals on these connectors either before or after a measurement. [Learn more.](#)

Test Set I/O

[See Details](#)

Bias IN and Fuses



Connect your DC Power Supply to apply Bias to the PNA ports through these BNC connectors.

- The bias fuses are rated for 0.5A. You are responsible to ensure that devices connected to the test port do NOT draw more current than 0.5A. This will occur, for example, if a calibration SHORT is connected to the test port with bias power ON. The fuse Agilent part number for the PNA-X is 2110-0824.
 - The PNA will meet all of its RF specifications with bias up to 200 ma. As the DC bias is increased, corrected source match and directivity will degrade at low RF frequencies.
-

Material Handler I/O

[See details.](#)

Pulse I/O

[See Details](#)

CPU

See [CPU Speed / Performance](#)

See [Determine Your PNA's CPU Version](#) (Internet connection required)

Removable Hard Disk Drive

See [Service Guide](#) to learn how to remove the HDD. (Internet connection required)

See [Preventing PNA Hard Drive Problems](#)

Last modified:

3-May-2012	Removed model references
19-Apr-2012	Updated RP image
28-Jun-2011	Added link to first time note
4-May-2011	Added N522x
12-Nov-2010	Added N5247A RF2
9-Apr-2009	Added new CPU board
14-May-2008	Added RF Path images
4-Sep-2007	Added 28V image
June 6, 2007	Added RF and IF connector images
January 11, 2007	MX New topic

Powering the PNA ON and OFF

The following is described in this topic:

- [How to...](#)
- [ON](#)
- [Shutdown](#)
- [Turn OFF Autostart](#)

Notes

During boot up of Windows or of the Network Analyzer application program, do **NOT** press keys on the front panel, rotate the RPG knob, or connect a USB device. Doing so **MAY** lead to a front panel lockup state.

If the PNA front-panel keypad or USB ports are not responding, **SHUTDOWN** or **RESTART** the PNA.

How to Log off, Shut down or Restart the PNA.

1. BRIEFLY press the front-panel PNA power button.
2. In the **What do you want the computer to do?** list, choose an action:
 - Log off (closes programs)
 - [Shut down](#)
 - Restart (shutdown and start)
3. Press **OK** to perform the action

Note: ONLY if the PNA is locked and you cannot operate the mouse or keypad - Press and hold the power button for at least four seconds. **This practice should be avoided!** Repeated shutdowns in this manner **WILL** damage the hard drive. [Learn more about damaging the PNA hard drive.](#)

ON Mode

- To turn ON the PNA press the power button.
- The power indicator will change to green when power is ON.

Turn OFF PNA Autostart

The PNA application (835x.exe) always starts automatically when power is turned ON. To cause the PNA to NOT Autostart, do the following:

1. Minimize the PNA application.
2. From Windows Explorer, navigate to and double-click the following file: C:/Program Files(x86)/Agilent/Network Analyzer/Service/Toggle_PNA_Autostart.exe

The script toggles the PNA Autostart mode ON and OFF.

Shutdown Mode

- In shut down mode the current instrument state is NOT automatically saved before the PNA is powered OFF.
- When the PNA is again powered ON, a full system boot-up is performed and the PNA powers-up in the [preset settings](#).
- A password may be required to resume PNA operation after being in Shutdown mode. [Learn more](#).
- To guarantee that your measurements meet the PNA specified performance, allow the PNA to **warm-up for 90 minutes** after the power indicator has turned green.
- The power indicator will change to yellow when power is OFF.

Note: If the PNA is locked and you cannot operate the mouse or keypad, shut down the PNA by pressing and holding the power button for at least four seconds.

This practice should be avoided! Repeated shutdowns in this manner WILL damage the hard drive. [Learn more about damaging the PNA hard drive.](#)

Unplugging the PNA

- Remove the power cord from the PNA ONLY when the power indicator is yellow, in either Hibernate or Shutdown mode. If the power cord is removed while the power indicator is green (PNA ON), damage to the hard drive is possible.
- The indicator will remain yellow for several seconds after the power cord has been removed.

25-Nov-2013	Updated folder location of toggle (win7)
3-May-2012	Removed hibernate
4-May-2011	Added N522xA models
3-Jun-2009	Updated for N5244A and N5245A

Traces, Channels, and Windows on the PNA

It is critical to understand the meaning of the following terms as they are used on the PNA.

- [Traces - Managing](#)
- [Channels - Managing](#)
- [Windows - Managing](#)

Other Quick Start topics

Traces

Traces are a series of measured [data points](#). There is no theoretical limit to the number of traces. However, the practical limit is the [maximum number of windows](#) times the maximum number of traces per window (24).

In addition, one memory trace can be stored and displayed for every data trace. [Learn more about Math / Memory traces](#).

Trace settings affect the presentation and mathematical operations of the measured data.

The following are Trace settings:

- [Parameter](#)
- [Format and Scale](#)
- [Smoothing](#)
- [Correction ON / OFF](#)
- [Electrical Delay](#)
- [Phase Offset](#)
- [Trace Math](#)
- [Markers](#)
- [Time Domain](#) (Opt 010)

Managing Traces

- [How to **Select** a trace](#)
- [How to **Delete** a trace](#)
- [How to **Move** a trace](#)
- [How to **Maximize** a trace](#)
- [How to **Create** a new trace](#)
- [How to **Change** the trace parameter](#)
- [How to display a custom **trace title**](#) (separate topic)
- [How to display a **wide** active trace](#) (separate topic)

How to Select a Trace

A trace must be selected (active) before its trace settings can be changed.

[How to know which trace is Active?](#)

Using front-panel

HARDKEY [softkey] buttons

1. For Traces 1-4, press the corresponding Hard Key
2. For other trace numbers, press **TRACES**
3. then [**Select Traces**]
4. Select a trace number in the [Entry toolbar](#).

PNA Menu using a mouse

1. Click the [Trace Status](#) label or trace.

How to Delete a Trace

Using front-panel

HARDKEY [softkey] buttons

1. For Traces 1-4, press the corresponding Hard Key
2. For other trace numbers, press **TRACES**
3. then **[Select Traces]**
4. Select a trace number in the [Entry toolbar](#).

PNA Menu using a mouse

1. Right-click the [Trace Status](#) label, then click Delete.

Programming Commands

How to Move a trace to a different window

You can **DRAG** a trace from one window to another, or...

Using front-panel

HARDKEY [softkey] buttons

1. Select the trace to move.
2. Press **TRACES**
3. then **[Move Trace]**
4. Select a window number in the following dialog.

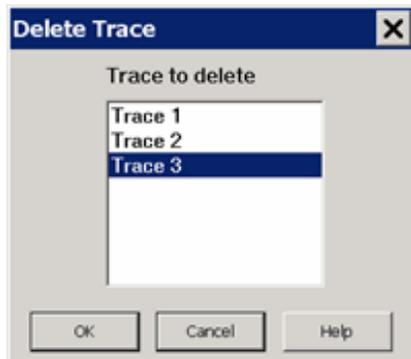
PNA Menu using a mouse

1. Right-click the [Trace Status](#) label, then click **Move Trace**.

Select, Delete, Move Traces dialog box help

This dialog is launched by clicking **Trace/Chan**, then **Trace**, then **Delete Trace**

The Select Trace dialog is launched by clicking **Trace/Chan**, then **Trace**, then **Select Trace**



Both the **Select Trace** and **Delete Trace** dialogs work the same.

Select a trace, then click **OK**.

Only ONE trace can be Selected or Deleted.

Note:

To EASILY select a trace, click the [Trace Status](#) label.

To EASILY delete a trace, right-click the Trace Status label, then click **Delete**.

Trace Max

Makes the active trace the ONLY trace on the display. All other traces are hidden.

How to do Trace Max

- Select **Trace**, then **Trace Max**.
- With Trace Max ON, select a different trace from the **Traces** softkeys to make that trace visible.
- To make all traces visible again select **Trace Max OFF**

Channels

Channels contain traces. The PNA can have up to **200 independent channels** (32 before A.09.00)

Channel settings determine **how** the trace data is measured . All traces that are assigned to a channel

share the same channel settings. A channel must be selected (**active**) to modify its settings. To select a channel, click the [Trace Status](#) button of a Trace in that channel. The following are channel settings:

- [Frequency range](#)
- [Power level](#)
- [Calibration](#)
- [IF Bandwidth](#)
- [Number of Points](#)
- [Sweep Settings](#)
- [Average](#)
- [Trigger](#) (some settings are global)

Managing Channels

How to Select a Channel

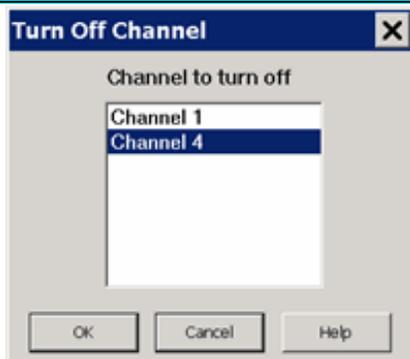
A channel must be selected (active) before its settings can be changed.

To make a channel active, [select a trace](#) in that channel.

How to Turn ON or OFF a Channel

Click **Trace/Chan**, then **Channel**, then **Turn On / Off** Channel.

Turn ON | OFF Channel dialog box help



Both the Turn **ON** and Turn **OFF** dialogs work the same.

Select a channel, then click **OK**. Only **ONE** channel can be selected

When turning ON a channel, the new channel is always the Standard [Measurement Class](#) with an

S11 trace.

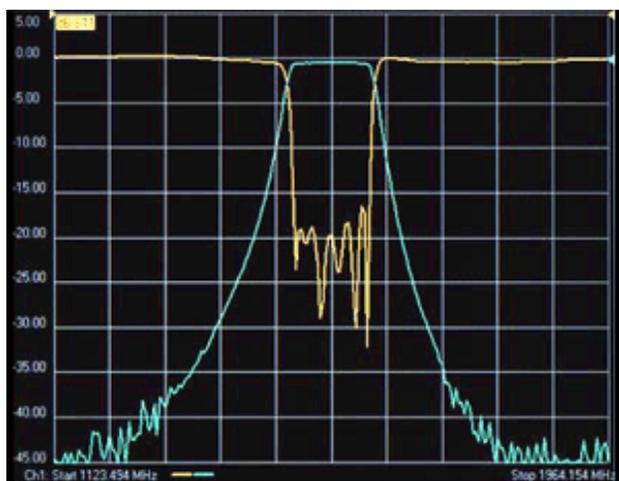
Note: To create more than one trace in a new channel, click Trace, then New Trace

Windows

Windows are used for viewing traces.

- The PNA can show an **UNLIMITED** number of windows on the screen (16 windows previous to PNA release 6.2) with the following limitations:
 - The COM property [MaximumNumberOfWindows](#) returns 1000 ('unlimited' is not a number).
 - The [SCPI status register](#) can only track the status of up to 576 traces.
- Each window can contain up to **24 traces** (8 traces previous to PNA release 8.2).
- Windows are completely independent of channels.
- Learn to [create and manage windows](#).
- See [Customize the PNA screen](#) to learn how to make other window settings.

The following is a window containing two traces. Both traces use the same channel 1 settings as indicated by the annotation at the bottom of the window.



PNA-X shows the window number in the lower-left corner of the window. The following shows window 5.



Managing Windows

How to make various window settings

New, Close, Tile, Cascade, Minimize, Maximize

Using front-panel

HARDKEY [softkey] buttons

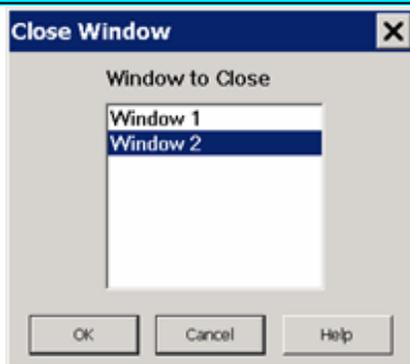
1. Press **RESPONSE**
2. then **[Display]**
3. then **[Windows]**

PNA Menu using a mouse

1. Click **Response**
2. then **Display**
3. then **Windows**

Programming Commands

Close Window dialog box help



Select a window, then click **OK**. The remaining windows are tiled.

Only ONE window can be selected.

Traces contained in a closed window are deleted.

Note: To EASILY close a window, right-click in the window (away from a trace) then select **Close Window**.

See [Customize the PNA screen](#) to learn how to make other window settings.

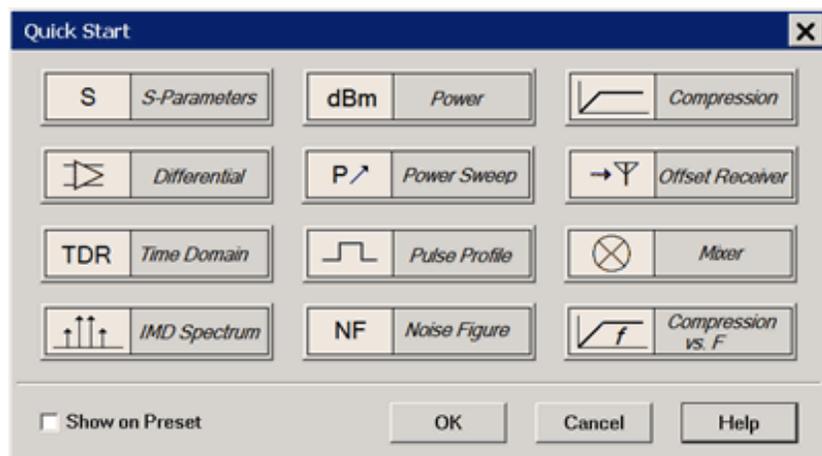
Last modified:

6-Apr-2012	Removed speed note
16-Mar-2012	Easy move trace
6-Dec-2011	Edit close window
21-Jul-2011	Moved some content to Customize.
4-Sep-2008	Removed legacy content
1-May-2008	Increased max number of traces per window
15-Oct-2007	MX New settings
9/19/06	MQ Modified for unlimited number of windows

Quick Start Dialog

Beginning with A.09.50, you can optionally see the following dialog when the PNA is Preset.

- Click a button to quickly setup the measurement type.
- The buttons appear ONLY if that option is installed on the PNA.



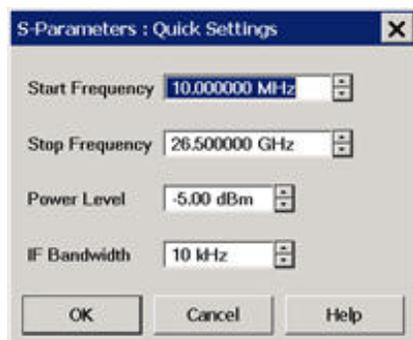
Click the image to learn more about each measurement.

- Clear **Show on Preset** to no longer see this dialog when Preset.
 - To see the dialog again, press **Preset**, then **[Quick Start]**.
-

S-Parameters

Required:
None

Creates S11 and S21 measurements in a single channel and window.



Enter:

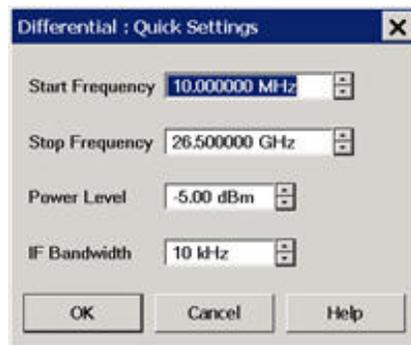
- [Start/Stop frequency](#)
- [Power Level](#)
- [IF Bandwidth](#).

Learn more about [S-parameter measurements](#).

Differential

Option
Required:
None

Creates Sdd11 and Sdd21 measurements in a single channel and window.



Enter [Start/Stop frequency](#), [Power Level](#), and [IF Bandwidth](#).

Learn more about [Differential \(Balanced\) measurements](#).

Time Domain

Option
Required:
010

Creates an S11 measurements and enables Time Domain.

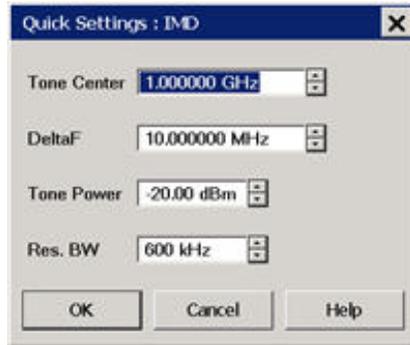


Enter [Start/Stop Time](#)

Learn more about [Time Domain measurements](#).

Creates an IM Spectrum measurement.

IMD
Option
Required:
087

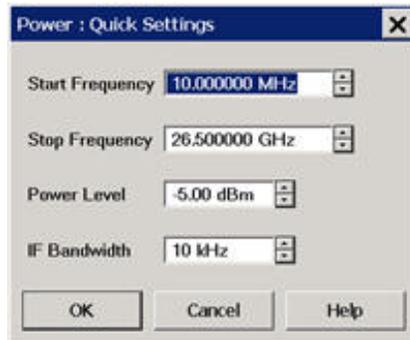


Enter [Tone Center](#), [DeltaF](#), [Tone Power](#), and [Res. BW](#)

Learn more about [IM Spectrum](#) measurements.

Power
Option
Required:
None

Creates R1 and B receiver measurements in a single channel and window. This allows you to view the DUT input power (R1) and output (B) power.

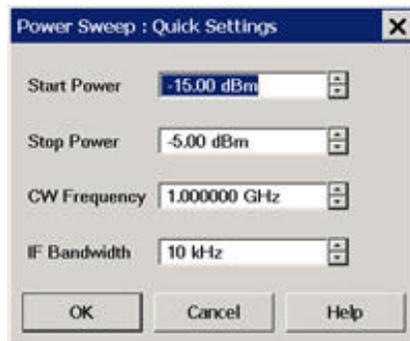


Enter [Start/Stop frequency](#), [Power Level](#), and [IF Bandwidth](#).

Learn more about [Unratioed Receiver measurements](#).

Power Sweep
Option
Required:
None

Creates a power sweep while viewing R1, B, and S21 measurements in a single channel and window. This allows you to view the DUT input power (R1), output power (B), and DUT gain (S21).



Enter [Start/Stop power](#), [CW Frequency](#), and [IF Bandwidth](#).

Learn more about [Power Sweep measurements](#).

Creates a pulsed stimulus at 1 GHz while viewing R1, and B receivers in a single channel and window. This allows you to view the pulse stimulus at the DUT input and output.

Pulse Profile

Option
Required:
008



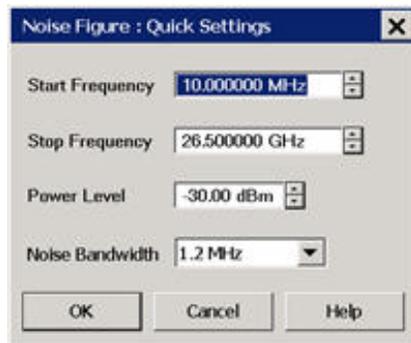
Enter [CW Frequency](#), [Power Level](#), Stop Time, [Pulse Width](#) and [Period](#).

Learn more about [Integrated Pulse Measurements](#).

Creates a standard NF measurement.

Noise Figure

Option
Required:
028/029



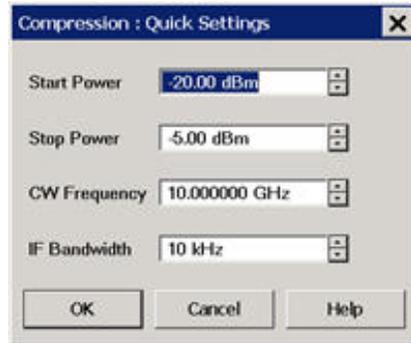
Enter [Start/Stop frequency](#), [Power Level](#), and [Noise Bandwidth](#).

Learn more about [Noise Figure measurements](#).

Creates a power sweep while viewing S21 (gain) with Compression Markers.

Compression

Option
Required:
None



Enter [Start/Stop power](#), [CW Frequency](#), and [IF Bandwidth](#).

Learn more about [Gain Compression Markers](#).

Offset Receiver

Option
Required:
080

Creates Frequency Offset Measurement while viewing R1 and B receivers in a single channel and window.



Enter Source [Start/Stop frequency](#) and [Power Level](#).

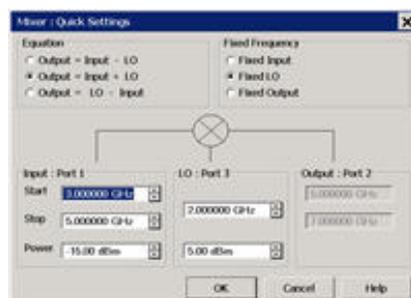
Enter Receiver [Start/Stop frequency](#) and [IF Bandwidth](#).

Learn more about [FOM](#).

Mixer

Option
Required:
082/083

Creates a Insertion Loss (SC21) measurement.



Enter Input, LO, and Output Frequencies and configuration.

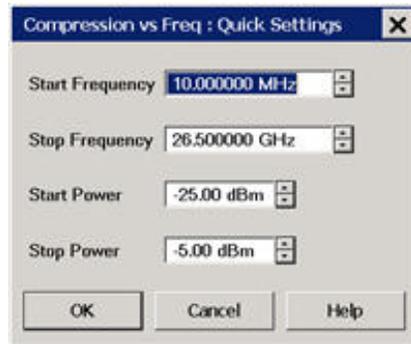
Learn more about [SMC Measurements](#)

Creates an CompOut21 (Gain Compression)

Compression vs Frequency

Option
Required:
086

measurement.



Enter [Start/Stop frequency](#), and Start/Stop [Power Level](#).

Learn more about [Gain Compression App](#)

Last modified:

23-Sep-2011 New topic

Basic Measurement Sequence

The following process can be used to setup all PNA measurements:

Step 1. Set Up Measurements

Reset the analyzer, create a measurement state, and adjust the display.

Step 2. Optimize Measurements

Improve measurement accuracy and throughput using techniques and functions.

Step 3. Perform a Measurement Calibration

Reduce the measurement errors by performing a calibration.

Step 4. Analyze Data

Analyze the measurement results using markers, math operations, and limit tests.

Step 5. Print, Save or Recall Data

Save or print the measurement data.

Frequency Blanking

For security reasons, you can prevent frequency information from appearing on the PNA screen and printouts.

How to set Frequency Blanking

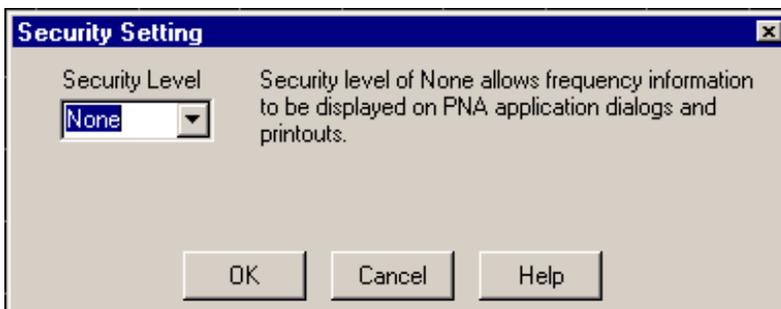
Using front-panel HARDKEY [softkey] buttons

1. Press **SYSTEM**
2. then [**Security**]

Using a mouse with PNA Menus

1. Click **Utility**
2. then **System**
3. then **Security**

← Programming Commands →



Security Setting dialog box help

Notes

- To learn how to erase memory before moving your PNA out of a secure area, see <http://na.tm.agilent.com/pna/security.html>.
- An ECal Data Wipe Utility destroys all user data per US DoD 5220.22-M. Learn more at <http://na.tm.agilent.com/pna/apps/applications.htm>
- PNA 'Undo' is disabled with **High** and **Extra** security levels. [Learn more](#).

Security Levels

None - All frequency information is displayed on the screen and printouts.

Low security level - Frequency information is blanked from the following:

- Display annotation
- Calibration properties
- All tables
- All toolbars
- All printouts
- [External sources](#) - See Also: [Preference to Deactivate External Devices on Preset](#). **Note:** Frequency Blanking is fully supported ONLY on Agilent MXG sources with option 006. On MXG models without option 006 and all PSG models, the window state is turned OFF. When the “local” button is clicked on the source, then frequency is re-displayed.

High security level - Low security level settings PLUS:

- [GPIB console](#) is inactive

Extra security level - High security level settings PLUS:

- All ASCII [data saving](#) capability (.snp, .prn, .cti) is saved without frequency information. The X-axis information is replaced with data point numbers. Before A.08.50, saving these file types was NOT allowed.
- [Mixer setup files](#) (*.mxr) can NOT be saved.

For ALL security levels:

Frequency information is **NOT** blanked from the following:

- [Service Adjustment Programs](#)
- Your COM or SCPI programs.

Instrument State and Cal Sets

The security level is always saved and recalled with an instrument state. However, the instrument state may contain a Cal Set or link to a Cal Set. [Learn more](#). This may influence the security level when the instrument state is recalled. Here is how.

- When a new Cal Set is created at the end of a calibration, the current system security level is stored with it.
- The only way to change an existing Cal Set's security level is by writing a new calibration into the Cal Set.

- When later applied to a channel, if the Cal Set has a **higher** security level than the current system security level, the system security level will become upgraded to that of the Cal Set.
- When saving an instrument state to either a *.csa or *.cst file, the security levels of the system and Cal Set are saved separately. When recalled, the higher security level of the two is applied.
- To view the security level of a Cal Set, see [Cal Set Properties](#).

Re-displaying frequency information

- When in **Low** security level, do any of the following:
 - Revisit this dialog box and select **None**
 - Perform an [instrument preset](#)
 - Recall an Instrument State/Cal Set with security level of **None**.
- When in **High** or **Extra** security level, do any of the following:
 - Perform an [instrument preset](#)
 - Recall an Instrument State/Cal Set with security level of **None**.

Last Modified:

8-Feb-2011	Clarified external sources
4-Nov-2010	Added external sources
21-Jun-2010	Clarified re-display freq information
17-Feb-2009	Added Extra file saving with data point numbers
4-Sep-2008	Removed legacy content
13-Aug-2008	Added ECal note
29-Apr-2008	Fixed hardkey presses
17-Jul-2007	Added Extra setting

Internal Second Source

The following PNA models include an internal second source.

Model	Total # of Ports
PNA-X Opt 224	<u>2</u>
ALL PNA-X models N522xA Opt 400's	4

How to use the second source

- Set frequency using the [Frequency Offset Opt 080 dialog](#).
- Set power using the [Advanced Power dialog](#).
- [Source power calibration](#) of the second source is performed as usual.
- Using FCA, [click the LO button](#) to set frequency and power.
- The [specifications](#) of the second source are the same as source 1.

Benefits / Uses of the second source

- Up to five times faster than stepping an external source.
- Measure Mixers with internal swept or fixed LO.
- Measure TOI or Intermodulation distortion.

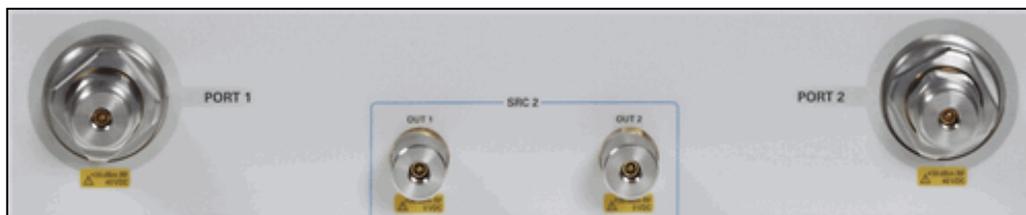
Internal Second Source Restrictions

Source 1 and Source 2 are available at specific ports as follows:

4-port models

- Source 1 power is available at Port 1 OR Port 2; NOT at both ports simultaneously.
- Source 2 power is available at Port 3 AND Port 4; BOTH ports simultaneously. (Although it is possible, the PNA firmware typically prevents both ports from sweeping simultaneously for measurement integrity purposes.)
- Other routing configurations are possible using the [RF Path Configurator](#).

PNA-X Opt 224 (PNA-X 2-port model):



- Source 1 power is available at **Port 1** OR **Port 2**; NOT at both ports simultaneously.
- Source 2 (**SRC 2**) power is available at **Out 1** AND **Out 2**; BOTH ports simultaneously.
- Other routing configurations are possible using the [RF Path Configurator](#).

Remotely Accessing the Internal Second Source

See [Remotely Specifying a Source Port](#).

Last modified:

7-May-2012	Removed N5230
4-May-2011	Modified for N522x
27-Apr-2011	All possibilities listed at Path Configurator
13-Nov-2010	Added port 1 combiner link.
26-May-2009	Added dual source path
6-Apr-2009	Replace N5242A with PNA-X
13-May-2008	Added remote section
1123-Jul-2007	MX Added PNA-X models
10/02/06	MQQ New topic

Networking and Connecting the PNA

The PNA as a PC

- [PNA User Accounts and Passwords](#)
- [Drive Mapping](#)
- [Using VNC to Control the PNA User Interface](#)

GPIB / COM Programming

- [Configure for COM/DCOM Programming](#)
- [Configure for GPIB, SCPI, and SICL](#)

Controlling External Devices

- [Configure an External Device](#)
- [E5091 TestSet Control](#)
- [External Testset Control](#)
- [Interface Control Feature](#)
- [TestSetIO Connector](#)
- [Handler IO Connector](#)

PNA Preferences

PNA preferences are settings that survive a Preset or PNA Shutdown. PNA Preferences are listed on this page with links to locations that provide more information.

How to set PNA Preferences

Using front-panel HARDKEY [softkey] buttons

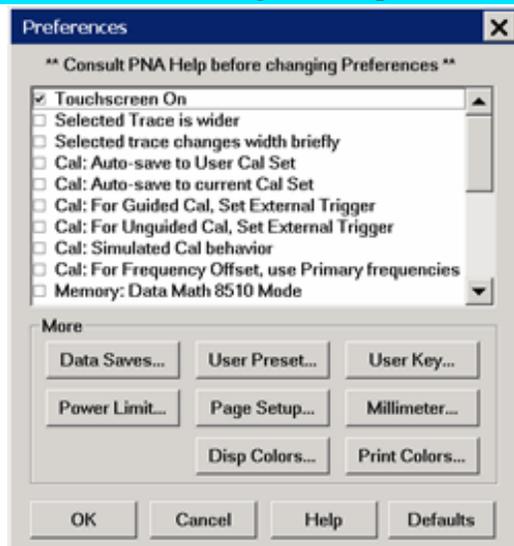
1. Press **SYSTEM**
2. then [**Configure**]
3. then [**More**]
4. then [**Preferences**]

PNA Menu using a mouse

1. Click **Utility**
2. then **System**
3. then **Configuration**
4. then **Preferences**

◀ Programming Commands ▶

Preferences dialog box help



PNA Preferences survive a Preset and a system reboot.

A checked box makes the following statements true unless stated otherwise.

Note: The default setting is listed first.

Touchscreen ON. Selections can be made by touching the screen.

Touchscreen ON. Selections can NOT be made by touching the screen.

Selected Trace is wider. The selected trace is the narrow, default size.

Selected Trace is wider. The active (selected) trace is always wider.

Selected trace changes width briefly. The selected trace does NOT change width briefly.

Selected trace changes width briefly.

This setting only affects calibrations performed using SCPI or COM. Cals performed from the User Interface ALWAYS offer a choice to save to a named Cal Set.

Cal: Auto-save to User Cal Set With Rev 6.0 we implemented a change that defaults to saving completed calibrations to Cal Registers instead of User Cal Sets.

Cal: Auto-save to User Cal Set Always automatically save completed calibrations to an auto-named User Cal Set. **Caution:** this can cause a lot of save User Cal Sets. [Learn more.](#)

The following message appears when both the Cal Set choices above and below are selected:

"Cal: Auto-save preferences conflict "

Auto-save to User Cal Set (above)- or - **Auto-save to current Cal Set.**(below)

Uncheck one of these.

This setting only affects calibrations performed using SCPI or COM. Cals performed from the User Interface ALWAYS offer a choice to save to a named Cal Set.

Cal: Auto-save to current Cal Set

Cal: Auto-save to current Cal Set Always automatically save a completed Cal to the Cal Set that is currently selected on the specified channel, which could be the channel Cal Register. If the channel does not yet have a selected Cal Set, the Cal will be saved to a new User Cal Set with an automatically-generated name.

Cal: For Guided (and unguided)Cal, set external trigger. With Rev 6.0 we implemented a change that allows the measurement of Cal Standards during a Guided Cal to be externally triggered when [trigger source](#) is set to External.

Cal: For Guided Cal (and unguided), set external trigger. Do NOT allow the measurement of Cal standard to be externally triggered. All Guided Cal Standard measurements are triggered internally regardless of the trigger source setting. [Learn more.](#)

Cal: For Unguided Cal, set external trigger.

Cal: For Unguided Cal, set external trigger. Same as above, except for Unguided Cals.

With A.09.50, this preference is no longer necessary. Use **For Guided Cal, set external trigger.**

For SCPI behavior only. [Learn more.](#)

Cal: Simulated Cal Behavior

Cal: For Frequency Offset, use Primary Frequencies

Cal: For Frequency Offset, use Primary Frequencies Use when making mmWave measurements without a test set. [Learn more.](#)

Memory: Data Math 8510 Mode Standard PNA data processing chain.

Memory: Data Math 8510 Mode Simulate the Agilent 8510 data processing chain as it pertains to Trace Math and Memory. [Learn more.](#)

Meas: Mathematical offset for receiver attenuation The reported test port receiver power is mathematically offset by the amount of receiver attenuation. Default for all models.

Meas: Mathematical offset for receiver attenuation The reported test port receiver power is NOT mathematically offset by the amount of receiver attenuation.

[Learn more.](#)

Meas: Mathematical offset for source attenuation The reported reference receiver power is mathematically offset by the amount of source attenuation.

Meas: Mathematical offset for source attenuation The reported reference receiver power is NOT mathematically offset by the amount of source attenuation.. [Learn more.](#)

Meas: RF power On during frequency sweep retrace The PNA leaves RF power ON during a retrace of single-band frequency or segment sweeps.

Meas: RF power On during frequency sweep retrace Turn RF power OFF during a retrace of single-band frequency or segment sweeps. [Learn more.](#)

Meas: Power Sweep Retrace At the end of a power sweep, while waiting to trigger the next sweep, the PNA maintains source power at the start power level.

Meas: Power Sweep Retrace Maintain source power at the STOP power level. [Learn more.](#)

Sets the scope of External Trigger Output signal properties. The PNA is **Preset** after changing this setting.

Meas: External Trigger OUT is Global Channels can have different External Trigger OUT settings. Default for PNA-X and N522xA models. On the Trigger Setup dialog, **Trigger Mode = Point** is ignored for external triggering.

Meas: External Trigger OUT is Global All channels have same External Trigger OUT settings. Default for PNA “C” and PNA-L models. Aux Trig OUT properties apply to all channels except the Per Point setting. To set Per Point for specific channels: On the [Trigger Setup](#) dialog, set **Trigger Scope** = Channel, under **Channel Trigger State**, select the channel, and set **Trigger Mode** = Point.

[See External Triggering dialog.](#)

Meas: Port 1 Noise Tuner Switch set to External The Noise Figure port 1 DPDT switch is set to EXTERNAL. This setting always provides incident power through the front panel loop.

Meas: Port 1 Noise Tuner Switch set to External The Noise Figure switch is set to INTERNAL, which is one method to make accurate S-parameter measurements when an ECal module is connected as a Noise Tuner.

Note: This preference is NOT available on PNA models with a Built-in Noise Tuner.

The Noise Figure App will throw the switch as needed. However, External Test Sets (in Multiport mode) rely on this switch being set to External. [Learn more about the Noise Tuner switch.](#)

Meas: Draw failed trace segments in red Failed data points (dots) are drawn in red.

Meas: Draw failed trace segments in red (default) Failed segments are drawn in red. [Learn more.](#)

For SCPI behavior only. [Learn more.](#)

Report source unlevelled events as errors Source unlevelled events are reported as errors.

Report source unlevelled events as errors Source unlevelled events are NOT reported as errors.

Ext Device: de-activate on PRESET and recall. External devices are de-activated when the PNA is Preset or when a Instrument State is recalled.

Ext Device: de-activate on PRESET and recall. External devices remain active when the PNA is Preset or when a Instrument State is recalled.

[Learn more about External Devices.](#)

On PRESET set two-point group delay aperture Group delay aperture is set to 11 points.

On PRESET set two-point group delay aperture Group delay aperture set to 2 points. [Learn more.](#)

On Preset turn power ON Instrument Preset always turns source power ON.

On Preset turn power ON When the current source power setting is OFF, source power remains OFF after Preset. When the current power setting is ON, source power is turned ON after

Preset. [Learn more.](#)

Turn Source Power Off when receiver is overloaded. Power remains ON when a receiver is overloaded.

Turn Source Power Off when receiver is overloaded. Turn OFF power to ALL ports when a receiver is overloaded. A notification dialog appears. Click **OK**, then lower the power level, then turn power ON. (Click **Stimulus**, then **Power**)

Report when receiver is overloaded A warning message is displayed on the PNA screen indicating that a receiver is overloaded or in compression. The displayed data is probably not accurate. One error per sweep appears and is reported in the [Error Log](#).

Report when receiver is overloaded Do NOT show overload warnings on the screen or report these errors in the error log.

The **More** buttons launch dialogs that contain predefined preferences:

Define Data Saves -

While not explicitly called Preferences, all of these settings survive a shutdown. [Learn more.](#)

User Preset

Specify the Instrument State file that the PNA will use when Preset. [Learn more.](#)

User Key

Sets softkey preferences. [Learn more.](#)

Page Setup

Standard printer settings (Paper, Orientation, and Size) do NOT survive a PNA shutdown.

All other settings DO survive a PNA shutdown. [Learn more.](#)

Millimeter settings

Sets MM Wave configurations. [Learn more.](#)

Power Limit

Sets Power Limits and Offsets. [Learn more.](#)

Disp Colors

Sets PNA display items to custom colors. [Learn more.](#)

Print Colors

Sets PNA print items to custom colors. [Learn more.](#)

Dialog Transparency

Some PNA dialogs can be viewed in various levels of transparency. [Learn more.](#)

Although they are called preferences, the following settings do NOT survive a PNA shutdown.

Calibration	UI Setting
Show or not, the first 'Method' Page of the Cal Wizard.	Cal Preferences
Set and order default Cal Types	Cal Preferences
Perform orientation of the ECal module during calibration?	ECal Wizard
Specify ECal port mapping when orientation is OFF	ECal Wizard
Show or hide custom Cal Windows during Cal	Cal Window (remote commands only)

Last Modified:

12-Aug-2013	Added dialog transparency (10.0)
18-Jul-2013	Reworded Noise tuner Switch pref
29-Apr-2013	Fixed receiver attenuation typo
14-Mar-2013	Red trace segment limit is default
11-Oct-2012	Edited for built-in noise tuner
5-Apr-2012	Added math to attn note
22-Feb-2012	Added note to Ext trigger for cals
29-Dec-2011	Added checkbox images
10-Sep-2010	Added Cal windows
26-Aug-2010	Added overload preferences
24-Mar-2010	Removed Error messages button
31-Jul-2009	Added External Device, primary FOM, draw failed segments, Power Limit, disp colors, print colors (9.0)
6-Feb-2009	Added new Cal set selection
22-Apr-2008	Added UI
10-Apr-2008	Added RemoteCalStorage and replaced AuxTrigger properties
5-Feb-2007	MX New topic

Using VNC to Control the PNA User Interface

VNC (Virtual Network Computing) allows you to control the User Interface of a PNA from any PC. The PNA display appears on the connected PC display. Mouse and keyboard control can occur from both the PNA and PC, although not simultaneously.

Note: For similar functionality, use Windows **Remote Desktop**. see the Windows help file for more information.

Both the PNA and PC must be connected to the same network. The responsiveness of the PNA while using VNC is dependent of the speed of your internet connection.

Every PNA is shipped with VNC installed. However, you must download and install the VNC software onto the PC.

The following procedures can help you configure VNC to view and control the PNA application from your PC.

On the PNA, run VNC Server

To do this:

1. Click View, then Minimize Application.
2. Click **Start**, then **All Programs**, then **TightVNC**, then **Tight VNC Server (Application Mode)**, then **Launch VNC Server**.
 - When the server is running, the  icon is visible in the lower right corner of the display. If not visible, click the up arrow to expand the list of icons.
 - The first time you run VNC Server, you must first set a password to control access from remote PCs.

On the PC, run VNC Viewer

To do this:

1. Download from <http://www.tightvnc.com/> and install TightVNC on the PC.
2. From the PC Desktop, click **Start**, then **Programs**, then **TightVNC**, then **TightVNC Viewer**
3. When prompted for the Hostname, type the [full computer name](#) or [IP address](#) of the PNA.
4. When prompted for the password, type the password you set when configuring VNC on the PNA.

25-Nov-2013 Updated for Win 7
15-Apr-2008 Updated for tightVNC

LXI-1.1 and VXI-11.3 Compliance

PNA-X, N522x, and PNA-C models that were shipped with PNA version A.08.20 or higher are LXI-1.1 and VXI-11.3 compliant.

LXI-1.1 Compliance

A PNA is LXI-1.1 compliant if the  logo appears on the PNA splash screen when the PNA application starts.

Learn more about LXI at <http://www.lxistandard.org/>

VXI-11.3 Compliance

To be compliant with VXI-11.3, the PNA must have been either:

- Shipped from the factory with PNA version A.08.20 or higher, or
- Had the Hard Disk Drive (HDD) upgraded since about June 2008 when A.08.20 was released and using PNA Rev. A.08.20 or higher.

Learn more about VXI at <http://www.vxi.org/>

LAN Status

When a LAN connection is used with the PNA, the LAN Status dialog allows you to see the IP address and other LAN connection properties.

How to view LAN Status

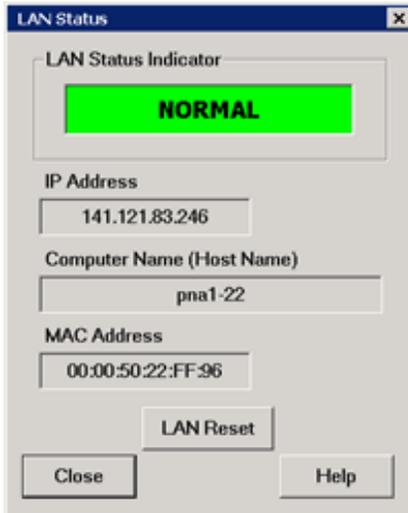
Using front-panel HARDKEY [softkey] buttons

1. Press **SYSTEM**
2. then **[Configure]**
3. then **[More]**
4. then **[LAN Status]**

Using a mouse with PNA Menus

1. Click **Utility**
2. then **System**
3. then **Configure**
4. then **More**
5. then **LAN Status**

 **Programming Commands** 



LAN Status dialog box help

Indicator Shows the current status of the LAN connection.

NORMAL - Indicates that the PNA LAN is ready for communication.

IDENTIFY - Indicates that a remote computer has invoked an LXI identification operation on the PNA using the web-based interface or [LXIDeviceIDState](#) COM property.

FAULT - Indicates that the PNA LAN interface is not connected to the Internet.

IP Address Shows the current IP address of the PNA

Computer Name Shows the full computer name of the PNA. [Learn how to change this.](#) If you see the IP address listed here, that means there is no DNS server specified in the network setup.

MAC Address Shows the unique address of the PNA computer. Also known as HostID.

LAN Reset Provides a LAN Configuration Initialize (LCI) mechanism. Press to return the following settings to factory default conditions:

- **IP Address Configuration (DHCP):** Enabled
- **ICMP Ping Responder:** Enabled
- **Web Password for configuration:** Resets the password for the factory default account (Agilent) to 'agilent'. If the **pna-admin** account had been renamed by the PNA owner, this function creates a new Administrator account named **pna-admin** and sets its password to 'agilent'. Learn more about [User names and Passwords.](#)

Web Server Software

If your PNA is LXI Class C compliant ([see above](#)), you can connect to the PNA using a web browser

over an internet connection.

To do this, when the above dialog indicates a **NORMAL** condition:

1. From a web browser, type **http://<your_PNA_computer_name>**. For example, to connect to the fictitious PNA in the dialog above, type: **http://pna1-22**
2. Type the log on User Name and Password
3. You will see the following welcome screen with connection links.

The screenshot shows the web interface for an Agilent Technologies PNA Series Network Analyzer. The page is titled "Welcome to your Web Enabled PNA" and features the LIXI logo. A sidebar on the left contains navigation buttons: Welcome Page, Print Page, Get Data, Links/Download/Upload/Reset, Send Remote Command, Send Remote Program, Shutdown/Restart, and View Configuration. The main content area is divided into two sections: "Information about this Web-Enabled PNA" and "Advanced Information about this Web-Enabled PNA".

Information about this Web-Enabled PNA :	
Description:	PNA Series Network Analyzer
Hostname:	study4-hana
IP Address:	141.184.84.160
VISA TCP/IP Connect String:	tcpip0::study4-hana:hpib7,16::instr
LXI Version:	1.1
LXI Class:	C
Toggle LXI Identification:	

Advanced Information about this Web-Enabled PNA:	
Model Number:	N5242A
Options:	400,P04,010,080,082,083,086,H08,423,551,020,021,022,025,014,081,502
Serial Number:	US46290061
Software Revision:	Z.08.10.10
Ethernet (MAC) Address:	00:00:50:29:B5:21
SCPI TCP/IP Socket Port:	5025
SCPI Telnet Port:	5024
SCPI Interface Name:	lan[141.184.84.160]hpib7,16
Auto-MDIX Capable:	No

© Agilent Technologies, Inc. 2007

Last modified:

- 4-May-2011 Added N522x
- 1-Apr-2011 Updated to include VXI

Windows 7 File Locations

With Windows 7, data file locations have been changed from previous Windows Operating Systems.

State files and most data file storage locations:

- Windows XP: c:\program files\agilent\network analyzer\documents
- Windows 7: c:\users\public\documents\network analyzer

PNA firmware executable file locations:

- Windows XP: c:\program files\agilent\network analyzer
- Windows 7: c:\program files (x86)\agilent\network analyzer

Support file locations

- Windows XP: c:\program files\agilent\network analyzer
- Windows 7: c:\programdata\agilent\network analyzer

These file locations can be queried remotely.

SCPI Command: [SYSTem:CONFigure:DIRectory?](#)

COM Command: [DirectoryPath Property](#)

See Also

[Your Programs on Windows 7](#)

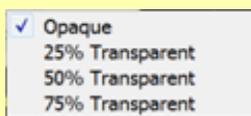
Dialog Transparency

Beginning with PNA release A.010.0, most PNA dialogs can be made to appear with various amounts of transparency. This allows you to view the PNA traces through the dialog as you make dialog settings.

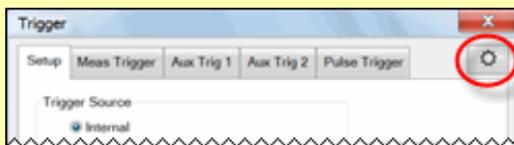
How to set Transparency Level

There are FOUR ways to make the transparency level setting:

1. Right-click in any non-control area of a dialog that allows transparency to see the following selections:



2. In tabbed dialogs, cycle through the above transparency settings by pressing the following button multiple times.



3. Launch the Transparency dialog (below) from the [Preferences](#) dialog.

4. Launch the Transparency dialog by doing the following:

Using front-panel

HARDKEY [softkey] buttons

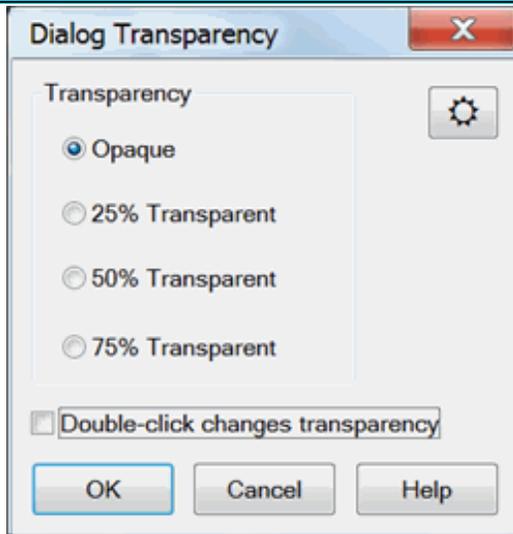
1. Press **DISPLAY**
2. then **[More]**
3. then **[Dialog Transparency]**

PNA Menu using a mouse

1. Click **Response**
2. then **Display**
3. then **Dialog Transparency**

This setting is not programmable

Dialog Transparency dialog box help



Note: This single Transparency setting applies to ALL supported PNA dialogs.

- Opaque (NOT Transparent) - Default setting
- 25% Transparency
- 50% Transparency
- 75% Transparency

Double-click changes transparency - When checked, cycle through the above transparency settings by double-clicking in any non-control area of a dialog that allows transparency.

Notes

- The transparency setting is stored as a [PNA Preference](#).
- The setting survives a PNA Shutdown and Preset
- It is NOT saved and recalled with instrument state.

Last Modified:

12-Aug-2013 New topic (10.0)

Using Help

Help Rev. March 12, 2014
PNA Rev. A.10.00
© Agilent Technologies, Inc.

This topic discusses the following:

- [PNA Documentation](#)
- [Printing Help](#)
- [Copying Help to your PC](#)
- [Launching Help](#)
- [Navigating Help](#)
- [Searching Help](#)
- [Help Languages](#)
- [Glossary](#)
- [Dialog Boxes](#)
- [Documentation Warranty](#)
- [Suggestions Please](#)

See Also

[Help, About Network Analyzer](#)

Other Quick Start Topics

PNA Documentation

This Help file, which is embedded in the PNA, is the **Users Guide and Programming Manual for the PNA**. The help file is automatically updated on the PNA when firmware is updated. Only the PNA [Installation and Quick Start Guide](#) is shipped with new PNA instruments.

Hardcopy manuals are no longer available for purchase with the PNA.

All PNA documentation, including the **latest online Web Help version** of this Help file, and a printable .PDF version of the Help file, are available at <http://na.tm.agilent.com/pna/help/index.html>.

Printing Help

Beginning with the PNA 5.2 release (March 2005), we once again offer a .pdf version of PNA Help. Download the .pdf file from <http://na.tm.agilent.com/pna/help/index.html>. You can still print individual PNA Help topics by clicking the Print icon at the top of the PNA Help window.

Copying Help to your PC

With the Help system on your PC, you can read about the PNA while away from it. You can also Copy and Paste programming code from this Help system directly into your programming environment.

The Help file is located on your PNA hard-drive at **C:/ Winnt/ Help/ PNAHelp.chm**. If both the PNA and PC are connected to LAN, you can [map a drive](#) and copy the file directly.

The Help file can also be downloaded from <http://na.tm.agilent.com/pna/help/index.html>.

Launching Help

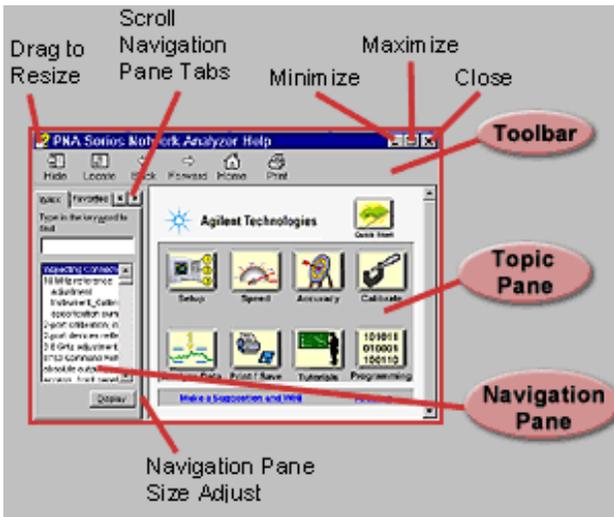
The Help system can be launched in three ways:

1. From the front panel Help button.
2. From the **Help** drop-down menu
3. From [Dialog Box](#) Help

Navigating Help

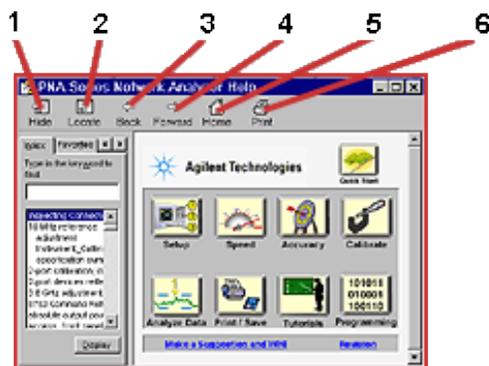
The Help Window contains 3 panes (regions):

1. [Toolbar](#) Pane
2. [Topic](#) Pane
3. [Navigation](#) Pane



Toolbar Pane

The Toolbar is at the top of all Help windows. It allows you to resize the window, browse and print the selected topic.



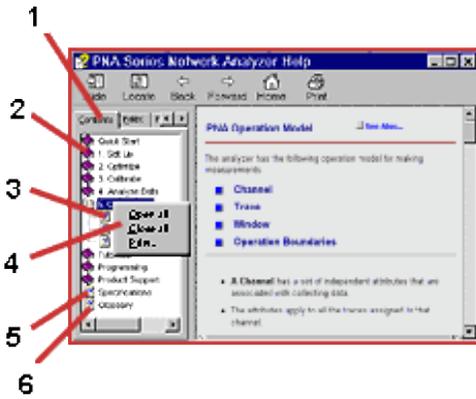
1. Hide or show the navigation pane
2. Locate the topic in the table of contents
3. Back to topic visited previously
4. Forward again if **Back** was clicked
5. Go to the Home page.
6. Print the topic pane.

Navigation Pane

Click the following tabs in the Navigation Pane to access information in the Help system:

- [Table of Contents Tab](#)
- [Index Tab](#)
- [Search Tab](#)
- [Favorites Tab](#)

(Table of)Contents Tab



1. Click tab to select Table of Contents.
2. Click a book to access related topics.
3. Click to display a topic.
4. Right click to access menu.
5. Click to display specifications
6. Click to display glossary

Index Tab

The index tab allows you to type a keyword and go to only the most applicable topics.



1. Click tab to select index.
2. Type keyword to find topics of interest.
3. View suggested topics. (Double-click to display topic.)
4. Click to display topic.

Search Tab

TIP: To Search any topic for a keyword, press **Ctrl** and **F**.

[Perform a Google search of the online version of PNA Help.](#)

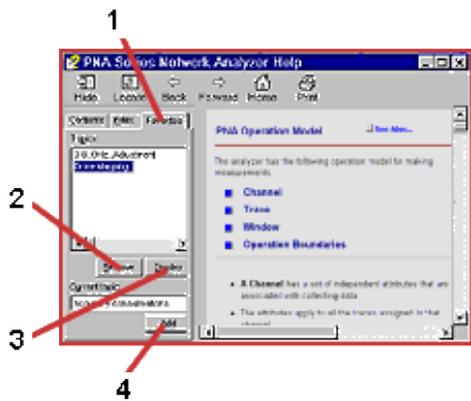
The following rules apply for using full-text search:

- Searches are not case-sensitive.
- You can search for any combination of letters (a-z) and numbers (0-9).
- Punctuation marks (period, colon, semicolon, comma, and hyphen) are ignored during a search.
- You can group the words of your search using double quotes or parentheses. Examples: "response calibration" or (response calibration). This requirement makes it impossible to search for quotation marks.
- Use Wildcard expressions:
 - To search for one undefined character use a question mark (?). For example, searching for **cal?** will find **calc** and **calf**.
 - To search for more than one undefined character use an asterisk (*). Searching for **Cal*** will find **calibration** and **calculate**.
- Use Boolean operators to define a relationship between two or more search words.

Search for	Example	Results will show topics containing:
Two words in the same topic	response AND calibration	Both the words "response" and "calibration".
Either of two words in a topic	response OR calibration	Either the word "response" or the word "calibration" or both.
The first word without the second word in a topic	response NOT calibration	The word "response" but not the word "calibration".
Both words in the same topic, close together.	response NEAR calibration	The word "response" within eight words of the word "calibration".

Favorites Tab

The favorites tab allows you to store (bookmark) the topics you refer to most often so that they can be recalled easily.



1. Click tab to view stored topics in Favorites.
2. Remove selected topic.
3. Display selected topic.
4. Add (store) current topic.

Topic Pane

The Topic pane allows you to view the contents of the selected topic.

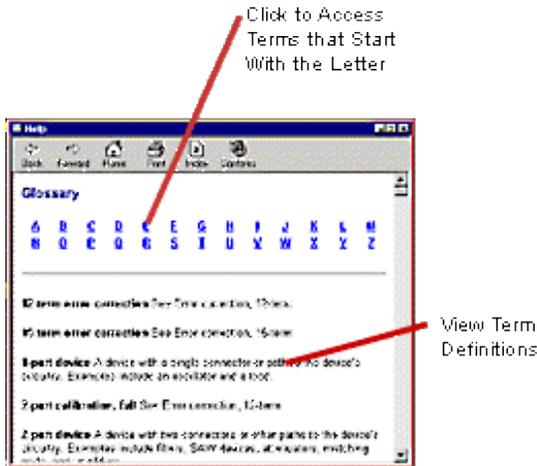


Help Languages

Beginning with PNA Rev A.08.00, PNA Help is offered in English ONLY.

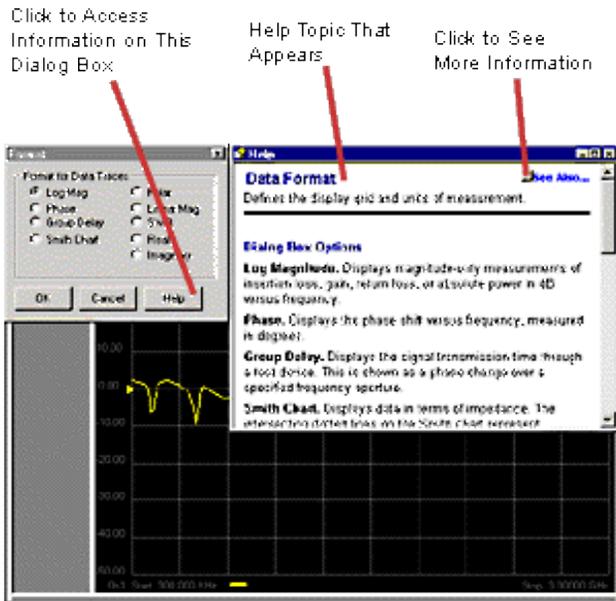
Glossary

The [Glossary](#) holds definitions of words, in alphabetical order.



Note: Click on a word in **green** text throughout Help to see the glossary definition.

Dialog Boxes



Documentation Warranty

THE MATERIAL CONTAINED IN THIS DOCUMENT IS PROVIDED "AS IS," AND IS SUBJECT TO BEING CHANGED, WITHOUT NOTICE, IN FUTURE EDITIONS. FURTHER, TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, AGILENT DISCLAIMS ALL WARRANTIES, EITHER EXPRESS OR IMPLIED WITH REGARD TO THIS MANUAL AND ANY INFORMATION CONTAINED HEREIN, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. AGILENT SHALL NOT BE LIABLE FOR ERRORS OR FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THE FURNISHING, USE, OR PERFORMANCE OF THIS DOCUMENT OR ANY INFORMATION CONTAINED HEREIN. SHOULD AGILENT AND THE USER HAVE A SEPARATE WRITTEN

AGREEMENT WITH WARRANTY TERMS COVERING THE MATERIAL IN THIS DOCUMENT THAT CONFLICT WITH THESE TERMS, THE WARRANTY TERMS IN THE SEPARATE AGREEMENT WILL CONTROL.

Suggestions Please!

Please let us know about your experience using PNA Help. Send your comments to:

pna_help@am.exch.agilent.com. Comment about any aspect of the help system. Here are a few areas that you might consider:

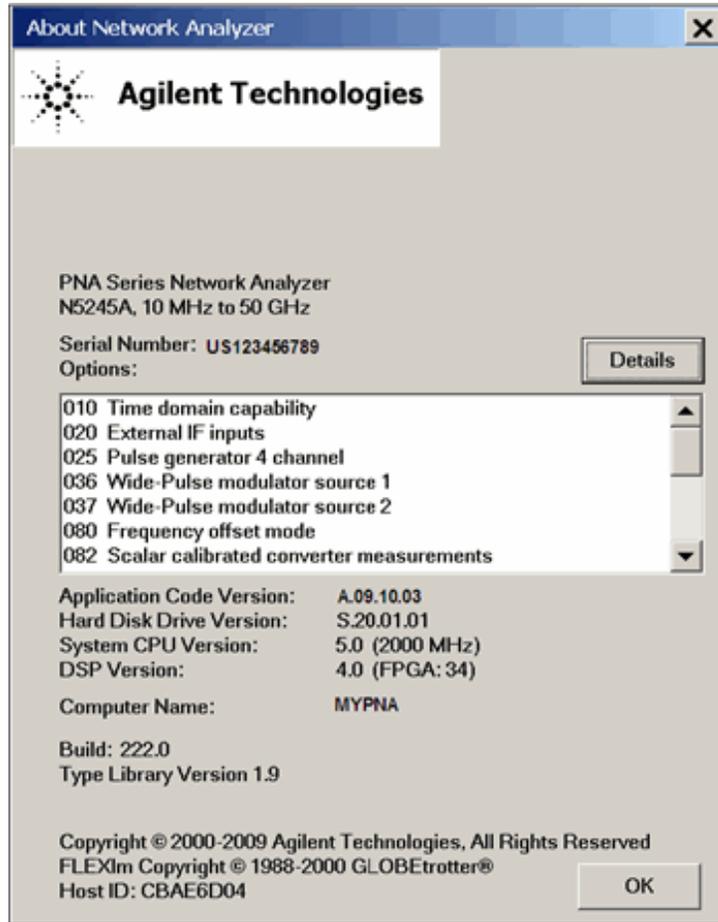
- Does anything appear to be broken?
- Did you find what you were looking for?
- Was the information you found helpful?
- Any suggestions as to how we can improve the help system?

Your comments go directly to the help system authors. For help with technical questions, please refer to [Technical Support](#).

Help - About Network Analyzer

Click **Help**, then **About Network Analyzer** to learn the following about your PNA:

Programming Commands



- Model number ([see list of PNA models](#))
- Frequency range
- Serial number
- Installed options ([Learn how to install software options](#))
- Application Code (firmware) Version
- Hard Disk Drive Version
- System CPU Version - [Learn more](#)
- DSP (Digital Signal Processor.) Version. [Contact Agilent](#) to upgrade the DSP. See DSP Version changes

below.

- Computer Name - [Learn more](#). This is also reported on the [LAN Status / LXI Compliance dialog](#).

DSP Version changes

For PNA-X, the DSP board number along with the FPGA version, provides access to the following features or causes the following settings to change:

DSP Board	FPGA Version	Changes
4	27	Enabled FastCW triggering
4	34	Enabled configurable (polarity and type) for PulseSyncIn external trigger signals.
5	xx*	<p>IF Frequencies – The default values are changed.</p> <p>IFAntiAliasFilter - The Narrow and Wide settings are changed.</p> <p>Note: Programs that control the following DSP filter settings, or state files that are saved with these settings, will yield different results when run or recalled on PNAs with DSP 4 versions versus DSP 5 versions.</p> <ul style="list-style-type: none">• Stage1 frequency (Max=38 MHz for DSP 5).• Stage2 Coefficient setting is IGNORED with DSP 5 Versions.• ADC Capture Mode Max data points increased to PNA Max.

*With DSP board 5, FPGA version is upgraded with firmware. This requires more time for FW installation.

Last Modified:

25-Aug-2010 Updated for DSP 5.0 (A.09.30)
30-Nov-2009 MX New topic

Search PNA Help

Search is limited to <http://na.tm.agilent.com/pna/help/latest/>



[See Google Search Help](#)

Preset the PNA

When you Preset the PNA, it is set to known, or preset conditions. You can use the factory default preset conditions, or define your own User Preset conditions.

- [Preset \(Default\) Conditions](#)
- [User Preset Conditions](#)

[See other 'Setup Measurements' topics](#)

Preset Default Conditions

How to Preset the PNA

Tip: Press the **Preset** button to start the PNA application if it is not already running.

Using front-panel HARDKEY [softkey] buttons

1. Press **Preset**

PNA Menu using a mouse

1. Click **Utility**
2. then **Preset**

[Programming Commands](#)

User Preset Conditions

The analyzer can be **preset** to either **factory default** conditions or **User Preset** conditions.

How to set User Preset

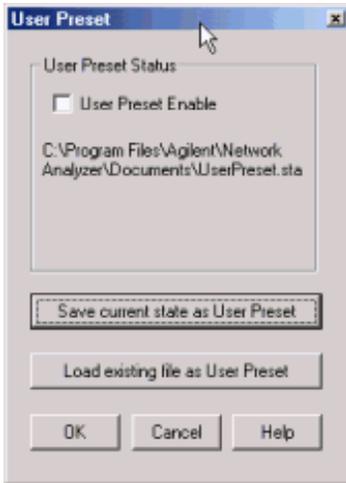
Using front-panel HARDKEY [softkey] buttons

1. Press **SAVE**
2. then **[User Preset]**

PNA Menu using a mouse

1. Click **Utility**
2. then **User Preset**

[Programming Commands](#)



User Preset dialog box help

With a User Preset saved and enabled, when the PNA is Preset, the User Preset settings are recalled instead of the factory default settings. Calibration data is NOT recalled with a User Preset. [Learn more about instrument state settings.](#)

User Preset Enable

Check - The PNA is preset to **User Preset** conditions when the Preset button is pressed.

Clear - The PNA is preset to **Default** conditions when the Preset button is pressed.

Save current state as User Preset Click to store the current instrument state as the User Preset conditions. File is stored as c:\users\public\network analyzer\documents\UserPreset.sta.

Load existing file as User Preset Click to retrieve an instrument state to be used as the User Preset conditions.

Last modified:

- | | |
|-------------|------------------------------------|
| 25-Nov-2013 | New path name for Win7 |
| 7-May-2012 | Removed most obsolete content |
| 9-Mar-2010 | Added link to Power ON/OFF |
| 3-Sep-2008 | Removed legacy content |
| 9/27/06 | MX Added UI |
| 9/12/06 | Added link to programming commands |

Measurement Classes

Measurement Classes are categories of measurements that can coexist on a channel.

- [What are Measurement Classes](#)
- [How to assign a Measurement Class to a Channel](#)
- [Measurement Class Dialog Box Help](#)

[See other 'Setup Measurements' topics](#)

What are Measurement Classes

The following table shows the Measurement Classes currently available for the PNA. Within each of these classes there are a number of measurements.

Measurement Classes are categories of measurements that can coexist on a channel. A measurement from one class can NOT reside in a channel with a measurement from another class. For example, a Noise Figure measurement can NOT reside in a channel that is currently hosting Scalar Mixer Measurements.

The Measurement Class dialog is accessed in the following ways:

How to assign a Measurement Class to a Channel

Using front-panel

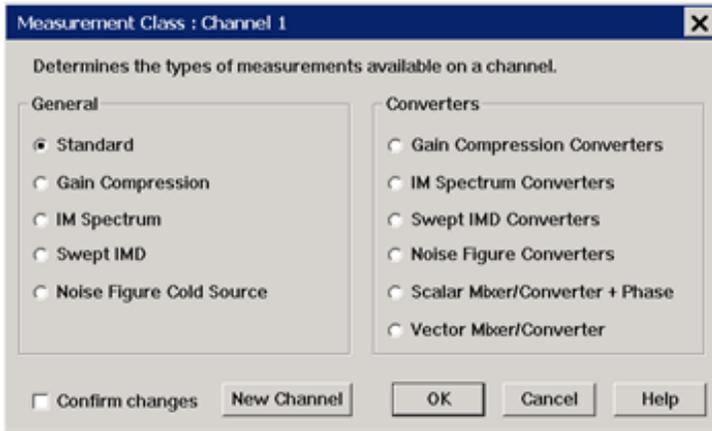
HARDKEY [softkey] buttons

1. Press **MEAS**
2. then [**Measurement Class**]

PNA Menu using a mouse

1. Click **Trace/Chan**
2. then **Measurement Class**

[Programming Commands](#)



Measurement Class dialog box help

Measurements in a measurement class can NOT coexist in a channel with a measurement of a different measurement class.

Select a measurement class for the active channel or new measurement channel.

- The **Standard** measurement class contains S-Parameters, Balanced parameters, and Receiver measurements.
- All other measurement classes are commonly called "[Applications](#)".

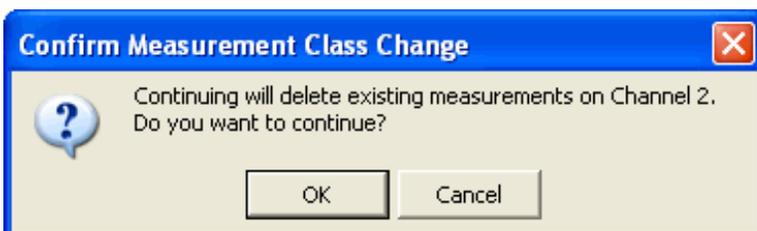
Title Bar Indicates the active channel to which the measurement class will be assigned.

Confirm changes

- Check (default setting) to launch the Confirm Measurement Class Change dialog.
- Clear to perform the 'OK' actions without confirmation. This setting survives a Preset and PNA Shutdown.

New Channel Click to create the measurement class in a new channel and new window. A default measurement for that class is created in the channel.

To change the measurement, click **Trace**, then select a new measurement.



Choose to do the following:

- **OK** - Delete the existing measurements in the active channel. Create the new measurement class, and default measurement, in that channel.
 - **Cancel** - Do not create the new measurement class. Leave the old measurements (and class) in that channel and return to the Measurement Class dialog box.
-

Last Modified:

29-Jul-2011	Several small DM changes
6-Aug-2010	Added Confirm changes
1-Jul-2010	Added Standard vs Application
22-Mar-2010	Updated dialog image
3-Sep-2008	Removed legacy content
18-Jun-2007	MX New topic

Measurement Parameters

This topic contains the following information:

- [S-Parameters](#) (pre-selected ratios)
- [Ratioed](#) (choose your own ratio)
- [Unratioed Power](#) (absolute power)
- [How to Select a Measurement Parameter](#)

[Learn about Balanced Measurements](#)

[See other 'Setup Measurements' topics](#)

S-Parameters

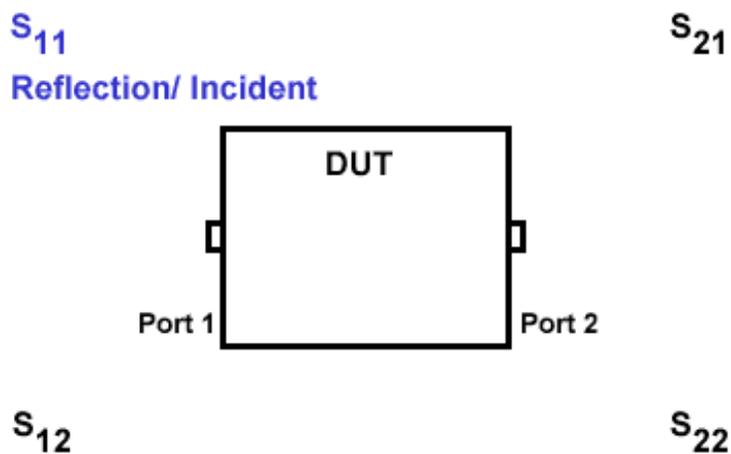
S-parameters (scattering parameters) are used to describe the way a device modifies a signal. For a 2-port device, there are **four S-Parameters**. The syntax for each parameter is described by the following:

S out - in

out = PNA port number where the device signal output is measured (receiver)

in = PNA port number where the signal is applied (incident) to the device (source)

Move the mouse over each S-parameter to see the signal flow:



For two-port devices:

- When the source goes into port 1, the measurement is said to be in the **forward** direction.
- When the source goes into port 2, the measurement is said to be in the **reverse** direction.

The analyzer automatically switches the source and receiver to make a forward or reverse measurement. Therefore, the analyzer can measure all four S-parameters for a two-port device with a single connection.

See the [block diagram](#) (including receivers) of your PNA.

Common Measurements with S-Parameters

Reflection Measurements (S11 and S22)

- Return loss
- Standing wave ratio (SWR)
- Reflection coefficient
- Impedance
- S₁₁, S₂₂

Transmission Measurements (S21 and S12)

- Insertion loss
- Transmission coefficient
- Gain/Loss
- Group delay
- Deviation from linear phase
- Electrical delay
- S₂₁, S₁₂

Receiver Measurements

All PNA models have test port receivers and reference receivers. See the [block diagram](#) of your PNA.

For 4-port models...

- R1, R2, R3, and R4 are reference receivers. They measure the signal as it leaves the PNA source.
 - R1 measures the signal out of Port 1
 - ...
 - R4 measures the signal out of Port 4
- A, B, C, and D are test port receivers. They measure the signal out (or reflecting off) of the DUT.

- A measures the signal into PNA Port 1
- B measures the signal into PNA Port 2
- C measures the signal into PNA Port 3
- D measures the signal into PNA Port 4

Receivers can also be specified using Logical Receiver Notation. [Learn more.](#)

Note: Beginning with PNA Rev. 7.22, you can use the internal ADC (Analog-Digital Converters) as measurement receivers. [Learn more.](#)

Ratioed Measurements

Ratioed measurements allow you to choose your own ratio of any two receivers that are available in your PNA. S-parameters are actually predefined ratio measurements. For example S11 is A/R1.

The following are common uses of ratioed measurements:

- Comparing the phase between two paths of a device. An example could be something simple like a power splitter or more complicated like a dual-channel receiver.
- Measurements that require a higher dynamic range than the analyzer provides with S-parameters.

Your PNA **MAY** have front-panel jumper cables that go directly to measurement receivers. Learn about the [front-panel jumpers](#) on your PNA.

Unratioed (Absolute Power) Measurements

The unratioed power parameter measures the absolute power going into any of the receivers that are available on your PNA.

The reference receivers are internally configured to measure the source power for a specific PNA port. Performing an absolute power measurement of a reference receiver using a different source port will measure very little power unless the front panel jumpers are removed and signal is applied directly to the receiver. An example of this would be an R1 measurement using port 2 as the source.

- [Measuring phase](#) using a single receiver yields meaningless data. Phase measurements must be a comparison of two signals.
- Averaging for Unratioed parameters is computed differently from ratioed parameters. [Learn more.](#)
- Choose **Unguided Cal**, then **Response Cal** if ONLY unratioed parameters are being measured. If S-parameters are also being measured, then SmartCal will calibrate the test port receivers in use.

How to create a NEW trace

The only measurements that can be created are those in the same measurement class as is currently assigned to the active channel. To create a measurement other than these, first assign the appropriate measurement class to a new or existing channel. [Learn how.](#)

Using front-panel

HARDKEY [softkey] buttons

1. Press TRACE 1, 2, 3, OR 4

PNA Menu using a mouse

1. Click Trace/Chan
2. then New Trace

Programming Commands

How to CHANGE the active trace

The only measurements that can be selected are those in the same measurement class as is currently assigned to the channel. To select a measurement other than these, first select the appropriate measurement class to a new or existing channel. [Learn how.](#)

1. Press MEAS
2. then select a new parameter

1. Click Response
2. then Measure
3. then select a new parameter

Programming Commands

New / Change Measurement dialog box help

Note: The only measurements that are available are those in the [measurement class](#) currently assigned to the active channel. Other measurements are NOT compatible.

To create a measurement other than these, first assign the appropriate measurement class to a new or existing channel. [Learn how.](#)

Click a tab to create or change measurements.

- When creating NEW measurements, you can choose more than one.
- When changing an EXISTING measurement, you can choose ONLY one.

Tabs

S-Parameter Select a predefined ratioed measurements. [Learn more about S-parameters.](#)

S-Parameter	Balanced	Receivers	
<input type="checkbox"/> S11	<input type="checkbox"/> S12	<input type="checkbox"/> S13	<input type="checkbox"/> S14
<input type="checkbox"/> S21	<input type="checkbox"/> S22	<input type="checkbox"/> S23	<input type="checkbox"/> S24
<input type="checkbox"/> S31	<input type="checkbox"/> S32	<input type="checkbox"/> S33	<input type="checkbox"/> S34
<input type="checkbox"/> S41	<input type="checkbox"/> S42	<input type="checkbox"/> S43	<input type="checkbox"/> S44

Balanced Select a balanced measurement type. (Multiport PNAs ONLY)

Change Click to invoke the [Balanced DUT Topology / Logical Port mappings](#) dialog box. [Learn more about Balanced Measurements.](#)

S-Parameter	Balanced	Receivers	
<input type="checkbox"/> Sss11	<input type="checkbox"/> Ssd12	<input type="checkbox"/> Ssc12	
<input type="checkbox"/> Sds21	<input type="checkbox"/> Sdd22	<input type="checkbox"/> Sdc22	
<input type="checkbox"/> Scs21	<input type="checkbox"/> Scd22	<input type="checkbox"/> Scc22	
<input type="checkbox"/> Imbal	<input type="checkbox"/> Sds21 Scs21	<input type="checkbox"/> Ssd12 Ssc12	
Topology / Mapping / Stimulus SE: 1 BAL: 2-3 Single End			<input type="button" value="Change"/>

Receivers Select receivers to make Ratioed and Unratioed (absolute power) measurements. [Learn more about receiver measurements.](#)

S-Parameter	Balanced	Receivers	
Activate: <input checked="" type="checkbox"/>	Numerator	Denominator	Source Port
	A	1.0	Port1
Activate: <input type="checkbox"/>	B	1.0	Port1
Activate: <input type="checkbox"/>	C	1.0	Port1
Activate: <input type="checkbox"/>	D	1.0	Port1
Activate: <input type="checkbox"/>	R	1.0	Port1
Activate: <input type="checkbox"/>	R1	1.0	Port1

Ratioed Check **Activate** to create or change a measurement. Select a receiver for the Numerator, select another receiver for the Denominator, then select a source port for the measurement.

The **Source port** is ALWAYS interpreted as a logical port number.

For convenience, the table is populated with common choices.

- [Learn about External Test Sets and Ratioed Measurements](#)
- [Learn more about Ratioed Measurements.](#)

Unratioed Same as Ratioed, but select **1** as the Denominator.

- [Learn More about Unratioed Measurements.](#)
- See the [block diagram](#) of receivers in YOUR PNA.
- The internal ADCs (Analog-Digital Converters) can be used as measurement receivers. [Learn more.](#)

Receiver Notation

With PNA [Rev 6.2](#), receivers can be also selected using logical receiver notation. This "8510-style" notation makes it easy to refer to receivers with an [External Test Set](#) connected to the PNA.

- **aN** - Reference receiver for logical port N
- **bN** - Test port receiver for logical port N

For example:

- For **Ratioed** measurements: "b12/a1" refers to the logical test port 12 receiver / the logical port 1 reference receiver.
- For **Unratioed** measurements: "b10" refers to the logical test port 10 receiver.

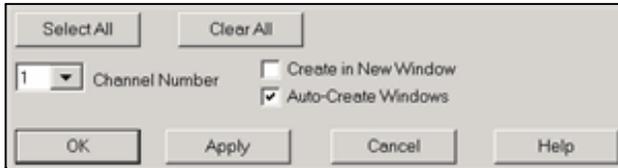
The PNA-style notation (A, B, R1 and so forth) can still be used to refer to the PNA **physical** receivers. [Learn more.](#)

However, ratioed measurements **MUST** use the same notation to refer to both receivers; either the physical receiver notation (A, R1) or the logical receiver notation (aN, bN). For example, the following mixed notation is **NOT** allowed: A/b3 and a5/R2.

Programming

When entering receiver letters using programming commands, neither logical or physical receiver notation are case sensitive.

Channel / Window Selections



These selections are NOT AVAILABLE when changing an EXISTING measurement. [Learn how to change a measurement.](#)

Channel Number Select the channel for the new traces.

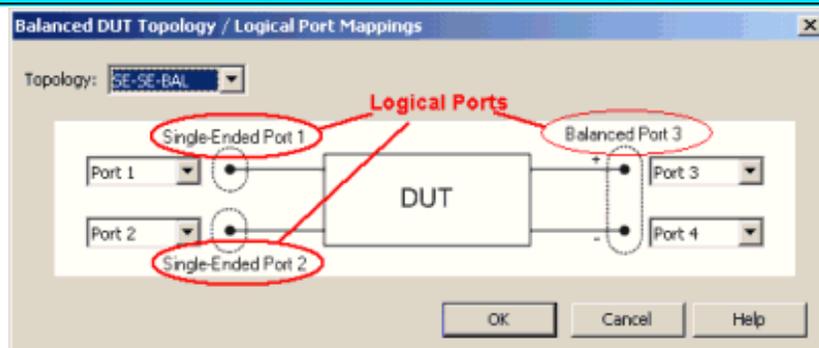
Create in New Window

- Check to create new traces in a new window.
- Clear to create new traces in the active window. When the PNA [traces per window limitation](#) has been reached, no more traces are added.

Auto-Create Windows Check to create new traces in as many windows as necessary. See PNA [number of windows limitation](#).

[About Measurement Parameters](#) (top of page)

Balanced DUT Topology / Logical Port mappings dialog box help



See this dialog for [Integrated True Mode Stimulus Application](#) (iTMSA).

Create or edit DUT Topology and Logical Port Mapping.

A Logical Port is a term used to describe a physical PNA test port that has been remapped to a new port number.

- Any **Two** physical PNA ports are mapped to **One Balanced** Logical port
- Any **One** PNA physical port is mapped to **One Single-Ended** Logical port

Note: These selections apply to ALL measurements in the channel. If the device topology is changed, any existing measurements in the channel that are incompatible with the new topology will be automatically changed to one that is compatible.

Topology: Describes your DUT as you would like it tested. The following device topologies can be measured by a multiport PNA.

- **Balanced / Balanced**
(2 logical ports - <4 actual ports>)
- **Single-ended / Balanced**
(2 logical ports - <3 actual ports>)
- **Single-ended - Single-ended / Balanced**
(3 logical ports - <4 actual ports>)

These topologies can be used in the reverse (<==) direction to measure:

- **Balanced / Single-ended** topology
- **Balanced / Single-ended - Single-ended** topology

For example, to measure a **Balanced / Single-ended** topology, measure the S12 (reverse direction) of a **Single-ended / Balanced** topology.

- Learn about [Logical Port mapping when using an External Test Set](#).
- Learn more about [Balanced Measurements](#)
- Balanced parameters can be saved to SNP files. [Learn more](#).

Last modified:

5-Feb-2013 Added unratioed Cal note
16-Mar-2012 Minor edits to notation
29-Apr-2011 Added images
2-Mar-2010 Link to SNP files
10/11/06 Added new UI
9/19/06 MQ Added logical receiver notation and Multiport meas toolbar.
9/12/06 Added link to programming commands

Frequency Range

Frequency range is the span of frequencies you specify for making a device measurement.

- [How to Set Frequency Range](#)
- [Zoom](#)
- [CW Frequencies](#)
- [Frequency Resolution](#)
- [Frequency Band Crossings](#)

[See other 'Setup Measurements' topics](#)

How to set Frequency Range

There are two ways to set the frequency range:

- A. Specify the **Start** and **Stop** frequencies of the range.
- B. Specify the **Center** frequency and desired **Span** of the range.

See the [frequency ranges of all PNA models](#)

Using front-panel HARDKEY [softkey] buttons

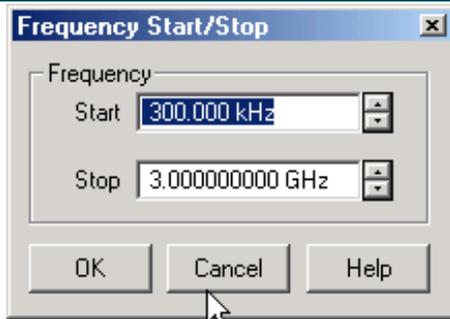
1. Press **FREQUENCY**

PNA Menu using a mouse

1. Click **Stimulus**
2. then **Frequency**

[Programming Commands](#)

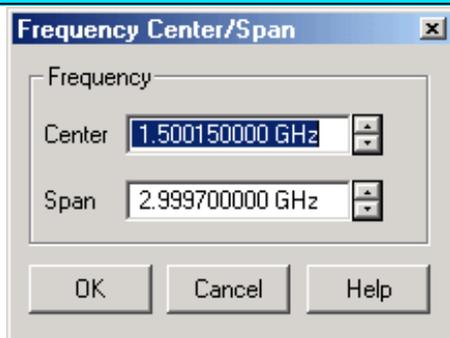
Frequency Start/Stop dialog box help



Start Specifies the beginning frequency of the swept measurement range.

Stop Specifies the end frequency of the swept measurement range.

Frequency Center/Span dialog box help



Center Specifies the value at the center of the frequency sweep. This value can be anywhere in the analyzer range.

Span Specifies the span of frequency values measured to either side of the center frequency.

Zoom

Zoom allows you to easily change the start and stop frequencies or start and stop power levels in a [power sweep](#).

Zoom operates on the [Active Trace](#) and all traces in the same channel as the active trace, regardless of the window in which they appear.

How to Zoom in a measurement window

1. Left-click the mouse or use a finger, then drag across a portion of a trace.
2. Release the mouse or lift the finger and the following menu appears:
3. Select from the following:
 - **Zoom** - changes the channel stimulus settings to the left and right border values of the Zoom selection
 - **Zoom xy** - changes the channel stimulus settings as above. In addition, the Y-axis scale of the active trace changes to the approximate scale of the Zoom selection.
 - **Zoom Full Out** - changes the channel stimulus settings to the full span of the current calibration. If no calibration is ON, then the stimulus settings are changed to the full span of the PNA model.

Notes

- The stimulus settings are changed for **ALL** traces in the active channel, regardless of the window in which they appear.
- If markers are in the selected area, they remain in place.
- If markers are in the unselected area, they are moved to the right or left edge of the new span. When Zoom Full Out is selected, the markers are moved back to their original location.

Zoom is NOT available for the following:

- Smith Chart or Polar [display formats](#)
- [CW Time](#) and [Segment sweep type](#)
- [Frequency Offset Measurements](#)
- [FCA Opt 083 Measurements](#)

CW Frequencies

Measurements with a [CW Time sweep](#) or [Power sweep](#) are made at a single frequency rather than over a range of frequencies.

How to set CW Frequency

1. Set [Sweep Type](#) to **CW Time** or **Power**.

You can also set CW frequency from within the Sweep Type dialog box.

Using front-panel

HARDKEY [softkey] buttons

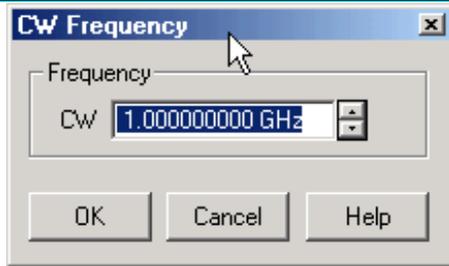
2. Press **FREQ**
3. then **[CW]**

PNA Menu using a mouse

2. Click **Stimulus**
3. then **Frequency**
4. then **CW Frequency**

◀ Programming Commands ▶

CW Frequency dialog box help



CW Type a value and the first letter of the suffix (k,m,or g) or use the up and down arrows to select any value within the range of the PNA.

Frequency Resolution

The resolution for setting frequency is 1 Hz.

Frequency Band Crossings

The frequency range of the PNA covers several internal frequency bands. The higher the frequency range of the PNA, the larger the number of bands. The source power to your DUT turns off as the stimulus frequency is swept through these band crossings. To learn more, see [Power ON and OFF during Sweep and Retrace](#).

The listed frequencies in the following tables are the stop frequency of the specified band, and the start

frequency of the following band.

You can download a [PNA Band Structure Readout](#) utility that lists the band crossings for your PNA.

Frequency band crossings are different for the following models:

- [PNA-X Models](#)
- [N522xA Models](#)
- [N523xA Models](#)

PNA-X (Band Stop Frequencies)

N5241A and N5242A

Band	Freq (GHz)	Band	Freq (GHz)	Band	Freq (GHz)
0	Reserved	12	.396	24	8.50
1	Reserved	13	.500	25	10.664
2	.014	14	.628	26	12.00
3	.019	15	1.00	27	12.80
4	.027	16	1.50	28	13.51
5	.038	17	2.00	29	15.40
6	.053	18	3.00	30	16.00
7	.075	19	3.20	31	18.00
8	.105	20	4.00	32	20.00
9	.146	21	5.332	33	21.328
10	.205	22	6.752	34	22.50
11	.250	23	8.00	35	24.00
				36	27.00

N5244A and N5245A

Band	Freq (GHz)						
0	Reserved	12	.396	24	8.50	36	27.008
1	Reserved	13	.500	25	10.664	37	32.00
2	.014	14	.628	26	12.00	38	36.50
3	.019	15	1.00	27	12.80	39	40.50
4	.027	16	1.50	28	13.51	40	42.656
5	.038	17	2.00	29	15.40	41	43.50
6	.053	18	3.00	30	16.00	42	46.20
7	.075	19	3.20	31	19.00	43	48.00
8	.105	20	4.00	32	20.00	44	50.20
9	.146	21	5.332	33	21.328		
10	.205	22	6.752	34	24.00		
11	.250	23	8.00	35	26.50		

N5247A

Band	Freq (GHz)	Band	Freq (GHz)	Band	Freq (GHz)
0	Reserved	12	6.75	24	32.00
1	Reserved	13	8.00	25	40.00
2	.053	14	10.70	26	40.50
3	.175	15	13.50	27	42.70
4	.250	16	15.40	28	46.20
5	.500	17	16.00	29	48.00
6	1.000	18	19.00	30	50.00
7	2.000	19	20.00	31	54.00
8	3.000	20	21.30	32	60.00
9	3.200	21	24.00	33	64.00
10	4.000	22	26.50	34	67.00
11	5.330	23	27.00	35	70.00

N522xA Models (Band Stop Frequencies)

N5221A and N5222A

Band	Freq (GHz)	Band	Freq (GHz)
0	Reserved	12	6.752
1	.010	13	8.0
2	.053	14	10.664
3	.175	15	13.51
4	.250	16	15.40
5	.500	17	16.0
6	1.0	18	20.0
7	2.0	19	21.328
8	3.0	20	24.0
9	3.2	21	27.0
10	4.0	22	
11	5.332	23	

N5224A and N5225A

Band	Freq (GHz)	Band	Freq (GHz)	Band	Freq (GHz)
0	Reserved	12	6.752	24	32.00
1	Reserved	13	8.00	25	40.50
2	.053	14	10.664	26	42.656
3	.175	15	13.51	27	43.50
4	.250	16	15.40	28	46.20
5	.500	17	16.00	29	48.00
6	1.000	18	19.00	30	50.20
7	2.000	19	20.00		
8	3.000	20	21.328		
9	3.200	21	24.00		
10	4.000	22	26.50		
11	5.332	23	27.008		

N5227A

Band	Freq (GHz)	Band	Freq (GHz)	Band	Freq (GHz)
0	Reserved	12	6.752	24	32.00
1	.010	13	8.00	25	40.00
2	.053	14	10.664	26	40.50
3	.175	15	13.51	27	42.656
4	.250	16	15.40	28	46.20
5	.500	17	16.00	29	48.00
6	1.000	18	19.00	30	50.00
7	2.000	19	20.00	31	54.00
8	3.000	20	21.328	32	60.00
9	3.200	21	24.00	33	64.00
10	4.000	22	26.50	34	67.00
11	5.332	23	27.008	35	70.00

N523xModels [See N523x Freq ranges.](#)

N5231A, N5232A, N5239A

Band	Freq (GHz)	Band	Freq (GHz)
0	Reserved	10	5.33
1	.000425	11	6.75
2	.0012	12	8.00
3	.0075	13	10.07
4	.025	14	13.50
5	.050	15	16.00
6	1.00	16	16.70
7	2.00	17	20.01
8	3.00		
9	4.00		

N5234A, N5235A Models

Band	Freq (GHz)	Band	Freq (GHz)	Band	Freq (GHz)
0		11	10.66	21	30.80
1	.010	12	13.51	22	31.50
2	.025	13	15.40	23	32.00
3	.050	14	16.00	24	40.00
4	1.00	15	20.00	25	40.56
5	2.00	16	20.26	26	42.00
6	3.00	17	21.33	27	48.00
7	4.00	18	24.00	28	50.00
8	5.332	19	26.00		
9	6.752	20	27.01		
10	8.00				

Last modified:

- 7-May-2012 Added N523x series
- 21-Jul-2011 Added N522xA series
- 9-Nov-2010 Added 5247A
- 9-Mar-2010 Made minor corrections
- 6-Apr-2009 Modified models
- 3-Sep-2008 Removed legacy content
- 10/23/06 MX Added new band crossings
- 10/16/06 Moved phase lock lost indicator
- 9/12/06 Added link to programming commands
- 9/27/06 MX Added UI

Power Level

Power level is the power of the PNA source at the test ports.

- [How to make Power Settings](#)
- [Power Dialog](#)
- [Power and Attenuator Dialog](#)
- [Source Unleveled](#)
- [Setting Independent Port Power](#)
- [Optimum Attenuation Value](#)
- [Receiver Attenuation](#)
- [Power ON and OFF during Save / Recall, User Preset, and Preset](#)
- [Power ON and OFF during Sweep and Retrace](#)

[See other 'Setup Measurements' topics](#)

Power Settings

The test port output power is specified over frequency. See the [Power Range specifications](#) for your PNA.

How to make Power settings

Use one of the following methods to set port power. Only the menu can be used to launch the [Power and Attenuators](#) dialog box.

Using front-panel HARDKEY [softkey] buttons

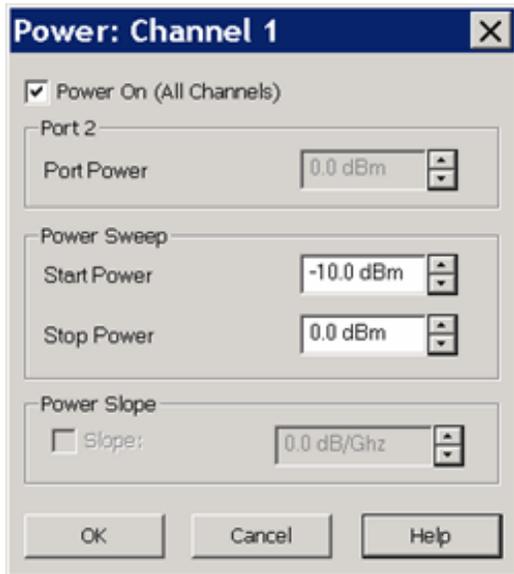
1. Press **POWER**
2. then **[Power]** or **[Power and Attenuators]**

Using a mouse with PNA Menus

1. Click **Stimulus**
2. then **Power** or [Power and Attenuators](#)

[Programming Commands](#)

[Power dialog box help](#)



Basic control of PNA source power for a specific port.

See [advanced control of source power and attenuation](#).

Power On (All Channels) Check to enable source power for all channels. Only turns power ON if channel power setting is ON or Auto. [See Advanced Power.](#)

Port 'n' Active source port for which power is being set.

Port Power Sets the power level for the specified port.

- To accurately set the power level at any point after the test port, perform a [Source Power Calibration](#).
- See the [specified power range of your PNA model](#).

Power Sweep

Start / Stop Power Set the start and stop power values of a power sweep.

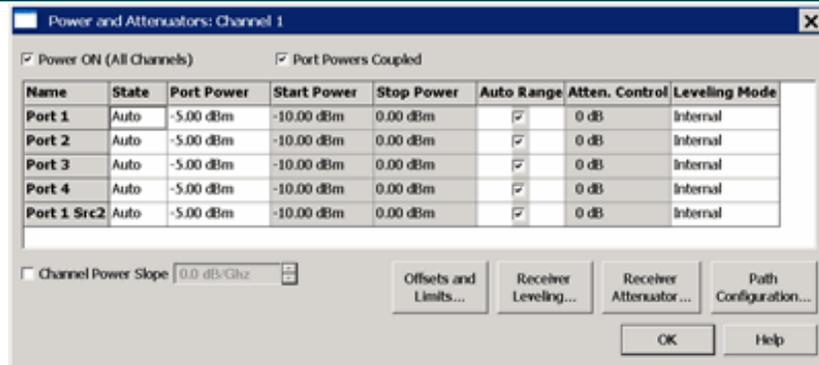
- These settings are only available when [Sweep Type](#) is set to Power Sweep.
- Uncoupled power sweep power can be set from the [Advanced Power dialog box](#).
- You can **Zoom** to easily change the start and stop power levels in a power sweep. [Learn how.](#)
- [Learn more about Power Sweep.](#)

Power Slope

Helps compensate for cable and test fixture power losses at increased frequency.

Slope Select to set the power slope. Clear to set power slope OFF. [Learn more about power slope.](#)

Power and Attenuators dialog box help



Defines and controls the PNA source power and attenuation for the active channel.

Beginning with PNA Rev. 7.21, external sources can be controlled from this dialog. [Learn more.](#)

Power On (All Channels) Check to enable source power for all channels. Only turns power ON if channel power setting is ON or Auto.

Port Powers Coupled

- **Coupled** (checked) The power levels are the same at each test port. Set power at any test port and all test ports change to the same power level.
- **Uncoupled** (cleared) The power levels are set independently for each test port. Uncouple power, for example, if you want to measure the gain and reverse-isolation of a high-gain amplifier. The power required for the input port of the amplifier is much lower than the power required for the output port. A power sweep can also be performed with uncoupled power. Learn more about [Setting Independent Port Power](#)

Name Lists the PNA test ports.

State

- **Auto** Source power is turned ON at the specified test port when required by the measurement. This is the most common (default) setting. See also [Power ON and OFF during Save / Recall, User Preset, and Preset.](#)
- **ON** Source power is ALWAYS ON, regardless of measurements that are in process. Use this setting to supply source power to a DUT port that always requires power, such as an LO port. This could turn OFF

power at another test port. [Learn about internal second source restrictions.](#)

- **OFF** Source power is never ON, regardless of the measurement requirements. Use this setting to prevent damage to a sensitive DUT test port.

Port Power Sets the power level at the output of the source.

- To accurately set the power level at any point after the test port, perform a [Source Power Calibration](#).
- See the [specified power range of your PNA model](#).
- See [ECal Module Compression Level](#)

Start / Stop Power Available ONLY when sweep type is set to Power Sweep. Set the start and stop power values of a power sweep. [Learn more about Power Sweep.](#)

- In PNA release 6.04 you can specify whether to maintain source power at either the start power or stop power level at the end of a power sweep. [Learn more.](#)
- In PNA release 8.20, a power sweep can be performed with [uncoupled power](#). Different power ranges can be swept in the forward and reverse directions.

Auto Range Check to allow the PNA to select the [optimum attenuation value](#) to achieve the specified test port power.

Clear to manually set the attenuation for each port. Type or select the attenuation value in the adjacent Attenuator Control box.

When using manual attenuation (Auto Range cleared), Port Power can be set within a 60 dB range. For example:

- With 0 dB of manual attenuation, Port Power can be set from -30 dBm to +30 dBm.
- With 10 dB of manual attenuation, Port Power can be set from -40 dBm to +20 dBm, and so forth.

Important Note: The available power range can also be adjusted AUTOMATICALLY by a Source Power Calibration, Guided Power Cal, or Power Compensation. If you are NOT seeing the range that you expect, or the correct power level at your DUT, view the Power Offset column in the [Power Limits and Offsets dialog](#).

Attenuator Control When Port Powers are Uncoupled, manual attenuator control allows you to set a wide range of power levels by setting the attenuation. [See Setting Independent Port Power.](#)

Also use manual attenuation control when a measurement requires a very good impedance match with the source, such as with oscillators or conditionally unstable amplifiers. Choose an

attenuation level of 10 dB or more to ensure the best source match.

- The PNA does not allow attenuators or other mechanical switches to switch continuously. [Learn more.](#)
- When Port Powers are Coupled, changing one port Attenuation Control value changes all port values.
- Attenuators are located between the source and the test port. Power to the reference receiver is not attenuated and is therefore higher than at the test port by the amount of attenuation. This will make uncalibrated measurements that use a reference receiver appear as though there is added attenuation at the test device. [See the PNA Block diagram.](#)

Note: Because the reference receiver is not in the attenuation path, there is more power at the reference receiver than at the test port by the amount of source attenuation.

By default, ALL PNA models mathematically offset the reported power at the reference receivers by the amount of source attenuation. [See Block diagram.](#)

With PNA release 7.2, a preference can be set to NOT mathematically offset the reported power of the reference receiver by the amount of source attenuation.

Learn how to [set the preference.](#)

Leveling Mode

- **Internal** Standard ALC leveling. Power level within an attenuator setting is limited to the ALC Range. [See Source Unleveled.](#)
- **Receiver Rx** Select a Reference receiver to use for leveling the source. [Learn more.](#)
- **Open Loop** (Used during pulse conditions with the internal source modulators). NOT available on [N523x models](#). No leveling is used in setting the source power. The lowest settable power, without attenuation, is limited to -30dBm. The source power level accuracy is very compromised. Use a source power calibration to make the source power somewhat more accurate.

Channel Power Slope

Helps compensate for cable and test fixture power losses at increased frequency. With power slope enabled, the port output power increases (enter positive value) or decreases (enter negative value) as the sweep frequency increases.

Slope Select to set the power slope. Clear to set power slope OFF.

Power slope is computed and applied from 0 GHz – not from the measurement start frequency. For example, with the following measurement settings:

- Start / Stop Freq: 10 GHz to 20 GHz
- Power level: 0 dBm
- Slope: 1 dB/GHz

The power into the DUT from 10 GHz to 20 GHz is 10 dBm sloping to 20 dBm

Offset and Limits Launches the [Power Offset and Limits](#) dialog.

Receiver Leveling Launches the [Receiver Leveling](#) dialog.

Receiver Attenuator Launches the [Receiver Attenuator](#) dialog.

Path Configurator Launches the [Path Configurator](#) dialog.

Source Unleveled

When the power level that is required at a test port is **higher** than the PNA can supply, a Source Unleveled [error message](#) appears on the screen and the letters LVL appear on the [status bar](#).

To perform a power sweep, the range of power is usually limited to the range of the Automatic Leveling Control (ALC) loop. (The PNA-X allows a very wide power range using **Open Loop**).

PNA specifications guarantee the ALC power range over which the PNA can supply power without an unleveled indication. However, the actual achievable power range on your PNA is probably greater than the specified range.

How to calculate the specified achievable power range

From the specifications for the frequency span from 15 GHz to 20 GHz:

- Max Leveled Power = **-8 dBm**
- Power Sweep Range (ALC) = **-17 dB**

For this frequency range the specified power range is calculated as:

- Max = **-8 dBm**
- Min = (-8)-(-17) = **-25 dBm**

When using Source Attenuators:

- with 10dB of attenuation, this becomes **-18 dBm** to **-35 dBm**
- with 20dB of attenuation, this becomes **-28 dBm** to **-45 dBm**, and so forth.

[See the output power specs for your PNA.](#)

To resolve an unlevelled condition, change either the Test Port Power or Attenuator setting. If an Unlevelled condition exists within the specified power range, [contact Technical Support](#).

Important Note: The available power range can also be adjusted AUTOMATICALLY by a Source Power Calibration, Guided Power Cal, or Power Compensation. If you are NOT seeing the range that you expect, or the correct power level at your DUT, view the Power Offset column in the [Power Limits and Offsets dialog](#).

Setting Independent Port Power

The PNA allows you to [uncouple port power](#) and specify different power levels at each test port. There are a few things to consider when setting independent port powers.

- Does your required high and low power levels fall within the specified Min and Max power range of the PNA? [See Unlevelled Indicator](#). If they do not, you may need to use the internal Source Attenuators.
- Does the PNA have source attenuators? If so, how many source attenuators? Some PNA models have one attenuator for each port. In most multiport PNA systems, the attenuators are shared by at least two test ports. See [PNA Options](#) to see the availability and range of source attenuation on your PNA.

Note: To prevent premature wear, the PNA does not allow attenuators or other mechanical switches to switch continuously.

These mechanical devices are set for the entire channel. When more than one channel is used, and a mechanical device setting is NOT the same for all channels, only the ACTIVE channel is allowed to sweep. All other channels are NOT allowed to sweep (Blocked).

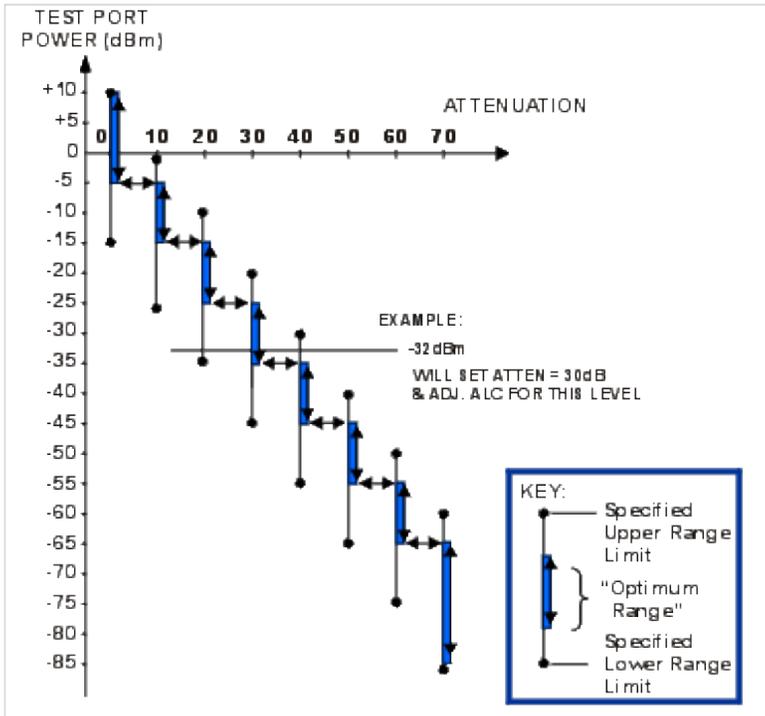
Learn how to [view the settings of all mechanical devices](#) in the PNA.

Optimum Attenuation Value

The attenuator has different positions, allowing a wide range of power levels. The number of power ranges available is determined by the source attenuation installed in your PNA. See [PNA Options](#) to see the availability and range of source attenuation on your PNA.

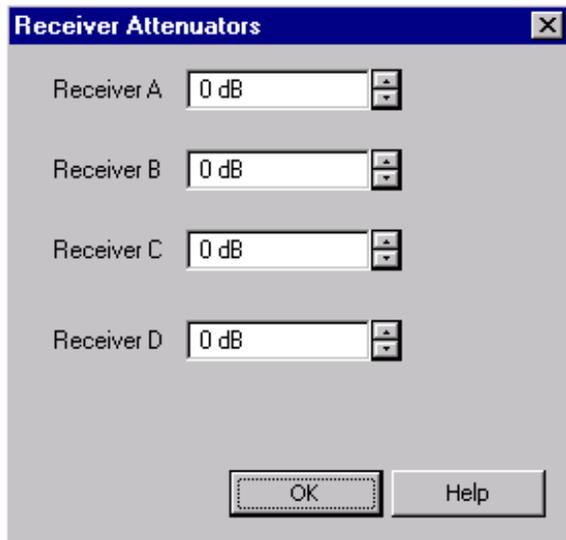
- Each range has a total specified span (25 dB in the following Attenuation Values graphic).
- The optimum setting is the middle of the range. This range provides the best accuracy and performance of the source leveling system. The optimum ranges are the blue regions in the following graphic.
- An attenuator setting can be selected manually or automatically. If automatic is selected, the blue optimum ranges (shown in the following graphic) are used.

(Attenuator ranges vary, this particular range is 70 dB)



Note: Error correction is fully accurate only for the power level at which a measurement calibration was performed. However, when changing power within the same attenuator range at which the measurement calibration was performed, ratioed measurements can be made with nearly full accuracy (non-ratioed measurements with less accuracy).

Receiver Attenuators dialog box help



Receiver Attenuators are offered as an option. [Learn more.](#)

Type or select independent attenuation values for each receiver.

- Receiver A is at Test Port 1
- Receiver B is at Test Port 2
- Receiver C is at Test Port 3
- Receiver D is at Test Port 4

Receiver Attenuation is used to protect the PNA test port receivers from damage or compression. Receiver attenuation causes the reported power at the receiver to be less than the power at the test port by the specified amount of attenuation.

Note: Beginning with PNA release 7.2, a preference can be set to mathematically offset (or NOT) the reported power at the test port receivers by the amount of receiver attenuation. By default, All PNA models offset the display.

Learn how to [set the preference.](#)

When an [external test set](#) is connected, Receiver Attenuation control is only available for the physical receivers in the PNA. Switching receiver attenuation using [logical receiver notation](#) is NOT allowed.

CAUTION! You can damage the analyzer receivers if the power levels exceed the maximum values.

- See [Technical Specifications](#) for the maximum input power to a receiver and receiver compression.

- See [Receiver attenuation values for your PNA model](#).

Power ON and OFF during Save / Recall, User Preset, and Preset

To protect your DUT from being inadvertently powered ON, the following RF Power ON/OFF settings occur:

[Instrument State Save/Recall](#)

If power is OFF when an instrument state is saved, then power will always be OFF after the instrument state is recalled.

If power is ON when an instrument state is saved, and the current power setting is OFF, then power will be OFF after the instrument state is recalled.

[User Preset](#)

If power is OFF when a User Preset is saved, then power will always be OFF after a User Preset.

If power is ON when a User Preset is saved, and the current power setting is OFF, then power will be OFF after a User Preset.

[Preset](#)

Instrument Preset sets power ON by default.

This can be changed with a [Preference setting](#) so that, if the current power setting is OFF, then power will be OFF after Preset.

Power ON and OFF during Sweep and Retrace

The frequency range of the PNA covers several internal frequency bands. The higher the frequency range of the PNA, the larger the number of bands. For example, a 9 GHz PNA has 6 frequency bands, a 50 GHz PNA has 25 frequency bands. See the [frequency band crossings](#).

Power to the DUT is turned OFF during band changes to avoid causing power spikes to the DUT.

Retrace occurs when the source gets to the end of your selected frequency span and moves back to the start frequency. Power to the DUT is again turned OFF when **retracing** across frequency bands.

Therefore, the following occurs for various stimulus settings:

1. **Single band sweep** - The power to the DUT is always ON, even during retrace. Beginning with PNA release 6.04, [a preference setting](#) can turn power OFF during a retrace. Only available in single band frequency and segment sweeps.

2. **Multi-band sweep** - The power to the DUT is turned OFF while sweeping across a band crossing. It is turned OFF again during retrace.
3. **Power sweep** - Because power sweep is always done at a single frequency, the frequency is always within a single band and the source power is always ON. At the end of a power sweep, power is immediately set to the start power.
Beginning with PNA release 6.04, this behavior can be changed with [a preference setting](#).
4. **Single sweep:**
 - Manual trigger mode - At the end of a multiband sweep, power is turned OFF during retrace, and then power is turned back ON before arming for the next trigger.
 - Hold mode - Power can be ON or OFF depending on when and how Hold mode is entered. However, power can be immediately turned OFF manually or remotely.

Caution: Avoid expensive repairs to your PNA. Read [Electrostatic Discharge Protection](#).

Last modified:

29-Apr-2013	Fixed recvr atten offset typo
3-May-2012	Removed C models
5-Apr-2012	Reword math offsets for attn
1-Dec-2011	Added Auto power offset note
15-Mar-2011	Open Loop Leveling not on C models
10-Mar-2011	Removed ON/OFF restriction
23-Feb-2010	Replaced Hold with Blocked channels
4-Aug-2009	Added Offset and Limits button
9-Feb-2009	Added Receiver Leveling
3-Sep-2008	Removed legacy content
27-Jun-2008	Added link to ECal compression
10-Jun-2008	Clarified power slope
28-May-2008	Updated for uncoupled power sweep
22-Apr-2008	Link to UI preferences
26-Mar-2007	Clarified retrace power OFF
11/16/06	Added new retrace features
10/23/06	Modified for new power diag
10/17/06	Clarified leveling
9/12/06	Added link to programming commands

Receiver Leveling

Receiver Leveling adjusts the source power until the measured receiver power is equal to the Port Power.

In this topic:

- [Overview](#)
- [Receiver Leveling Process](#)
- [Features and Limitations](#)
- [How to make Receiver Leveling settings](#)
- [Receiver Leveling dialog box help](#)
- [Initial Power Selection](#)

See other '[Setup Measurements](#)' topics

Overview

Receiver Leveling uses PNA receiver measurements to adjust the source power level across a frequency or power sweep. Before each measurement sweep, a variable number of background sweeps are performed to repeatedly measure power at the receiver for each stimulus point. Those power measurements are then used to adjust the source power level and achieve greater source power level accuracy.

This is similar to a [Source Power Calibration](#) which makes a **single sweep** to measure source power. The source power correction values are applied for all subsequent measurement sweeps. Because Receiver Leveling is performed for **every measurement sweep**, it provides more accurate source power levels, but also takes longer to perform each measurement sweep.

Receiver Leveling Process

Leveling sweeps are performed in the background (not visible) before every measurement sweep to measure and apply source correction data.

1. For each leveling sweep, source power is applied at each data point and measured by the specified receiver. [Learn how the initial power level is selected.](#)
2. The deviation is calculated between the measured power and the port power.

3. The deviation is applied to the current source power, and the updated source power levels are applied on the following leveling sweep.
4. This process continues until the receiver power at each data point has achieved the port power within the specified tolerance value, or until the specified number of leveling sweeps (iterations) has been reached.

Features and Limitations

- Receiver Leveling can be used with most [sweep types](#), including Segment sweep and Power sweep. See [Wide Power Sweep with Receiver Leveling](#).
- Receiver Leveling is ALWAYS enabled for the controlled source when [Phase Control](#) (Opt 088) is enabled.
- Receiver Leveling is available for standard S-parameter measurements and with [FCA](#), [GCA](#), and [IMD](#) applications.
- Turn ON Receiver Leveling **before** or **after** doing a Calibration. When turned ON before calibrating, it is turned OFF during the calibration, then back ON after calibration.
- Power Offset on the [Offsets and Limits dialog](#) can be used when there exists an additional attenuator or booster amplifier in the source path. An offset should be set to improve the leveling speed. This power offset is automatically used to set the port power.

Use Receiver Leveling for the following:

- Correcting for short term drift when using an external component, such as a booster amplifier. The booster amplifier must be connected to the front-panel jumpers, in front of the reference receiver. See the Block diagram for your PNA, located at the end of every [Specifications document](#).
- Extending the accuracy of power leveling at very low powers where the internal detector may be too noisy.
- Providing controlled power during [Pulsed measurements](#) in an open loop mode.
- Controlling the power at the outputs of [MM-Wave heads](#).

How to make Receiver Leveling settings

Start the [Power and Attenuators](#) dialog box as follows:

Using front-panel HARDKEY [softkey] buttons

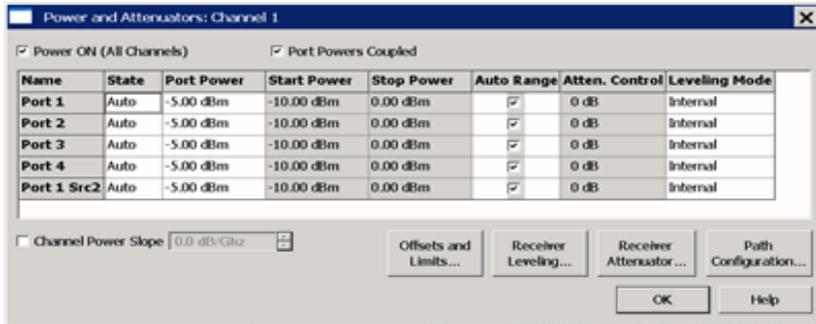
1. Press **POWER**
2. then [**Power and Attenuators**]

Using a mouse with PNA Menus

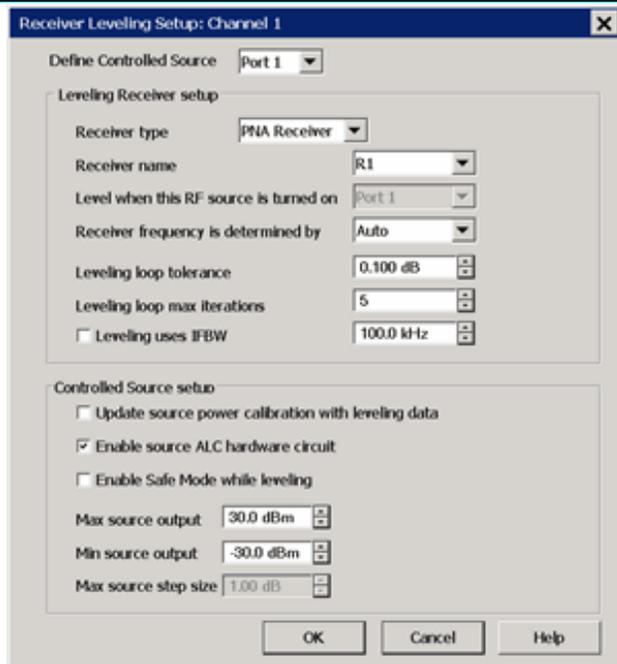
1. Click **Stimulus**, then **Power**
2. then [Power and Attenuators](#)

 [Programming Commands](#) 

3. On the Power and Attenuators dialog, press **Receiver Leveling**.



Receiver Leveling dialog box help



[Learn about Receiver Leveling](#) (scroll up).

Define Controlled Source (Port)

Each source port to be leveled is configured individually. Select a source to be configured for receiver leveling. Choose from: Port 1, Port 2, Port 3, Port 4, or any active external RF or DC source. [Learn more about External Devices](#).

Leveling Receiver Setup

Receiver Type Receiver type does an initial sort to make it easier to select a receiver. Choose from: PNA Receiver or Ext. Device ([PMAR](#)).

Receiver Select a receiver to be used to level the specified source.

For PNA Receiver type, choose from any PNA receiver using standard or receiver notation.

To level power at the source output or DUT input choose the reference receiver for the source port. For example, to level the source power at port 1, then choose "R1". To level power at the DUT output, choose the receiver that is used to measure the DUT output. If the DUT output is connected to port 2, then select "B" or "b2". [Learn about Receiver Notation.](#)

When [Phase Control](#) is enabled, the ratioed receivers used in Phase Control are selected and can NOT be changed. However, the Reference Source CAN also be selected for Receiver Leveling.

For Ext Device type, choose a configured PMAR device.

Level when this RF source is turned ON: The Controlled Source is selected automatically and can NOT be changed.

Receiver frequency is determined by: Available ONLY when the selected receiver is a PNA Receiver or power meter. This setting determines which receiver frequencies are measured. Choose from:

- **Auto** - always uses the frequency range that is assigned to the measurement receiver.
- **FOM Receiver** - FOM Receiver frequency range. Learn more about [Frequency Offset Mode.](#)
- **FOM Source** - FOM Source frequency range.
- **DUT Input** - Mixer/Converter input frequency range.
- **DUT Output** - Mixer/Converter output frequency range.

Leveling Loop Tolerance The source is considered leveled when each stimulus data point has achieved the power level +/- (plus or minus) this tolerance value.

Leveling Loop Max Iterations If every stimulus data point does NOT achieve the port power after this number of leveling sweeps, the measurement sweep occurs using the correction values obtained from the last leveling sweep. The message: **Not settled, noisy trace** appears when the Max Iterations is reached. If you see this message, you can increase the Max Iterations, reduce the IFBW, or increase the Tolerance setting.

Leveling IFBW Available only for PNA receivers. By default, the IFBW for the leveling sweeps is set to 100 kHz. [Learn more about IFBW.](#)

- Increase this value to make faster, but noisier leveling sweeps.

- Decrease this value to make slower, more repeatable leveling sweeps.
- Uncheck the box to use the same IFBW as the measurement sweeps.

Controlled Source Setup

Update source power calibration with leveling data Available only when using an RF Source and PNA receiver.

- When checked, the latest correction data is copied to the Source Power Cal correction array. When Leveling Mode is switched back to Internal (on the Power and Attenuators dialog), [Source Power Cal](#) is automatically turned ON using this correction data.
- When cleared, Source Power Cal is NOT turned ON when Leveling Mode is switched back to Internal.

Enable Source ALC hardware circuit Available only when a PNA source is selected.

- When checked, PNA internal leveling hardware is used. (Recommended)
- When cleared, [Open Loop hardware](#) is used. NOT available on [N523x models](#).

Enable Safe Mode while leveling

To protect your DUT, these settings control the extent to which the source power will be changed to achieve the port power as measured at the reference receiver. These settings could be necessary when using external components with a large variation in frequency response (flatness).

When checked:

- The Min source output is used as the initial power level for the leveling loop process.
- The controlled source is never stepped more than the Max source step size.

When cleared:

- The initial power for the leveling loop may be determined by the Min source output, the Max source output, the last setting of the leveling loop, or the target value of the leveling loop. [See Initial Power below](#).
- The Max source step size is ignored.

Max source step size When Safe Mode is enabled, the change in source power at each data point from one sweep to the next is limited to this value. For example, assume Safe Mode is enabled,

and Max Power Step is set to 1 dB. On the first leveling sweep, the first data point measures 3 dB lower than the port power, then source power for data point 1 will be increased by 1 dB for the next sweep, and likely for the following two sweeps.

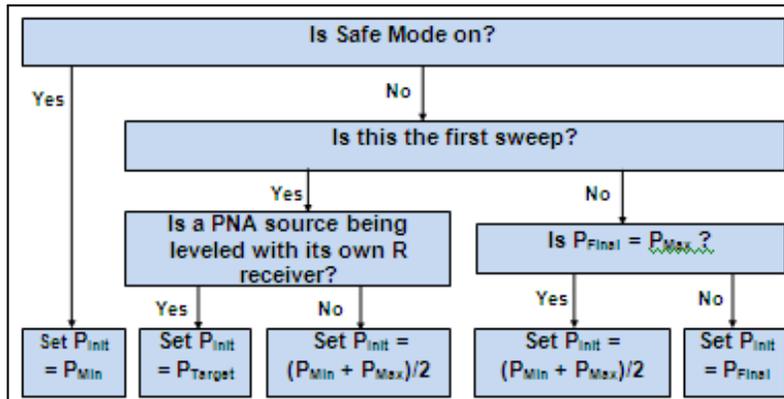
Max source output Always limits the maximum power out of the source to this value. The message: **Power set to Max Power** appears when this limit is reached.

If the maximum port power out of the PNA is reached at any time during the leveling sweeps, the following message appears: **Power set to user power limit.**

Min source output Always limits the minimum power out of the source to this value. The message: **Power set to Min Power** appears when this limit is reached. When Safe Mode is enabled, this value is used as the initial power level for the leveling loop process

Initial Power Selection

For each displayed data point, the leveling algorithm must select an initial power to begin the iteration process. This value is chosen as follows:



Where:

P_{Init} = the initial power for the iteration process.

P_{Final} = the final power setting from the previous leveled sweep.

P_{Min} = the minimum controlled source output level as specified in the Receiver leveling setup.

P_{Max} = the maximum controlled source output level as specified in the Receiver leveling setup.

P_{Target} = the target power level for the selected leveling receiver.

22-Jul-2013	Fixed typo
3-May-2012	Removed C models
7-Sep-2011	Significant changes with A.09.50
12-May-2011	Added before or after cal
15-Mar-2011	Added ratioed receivers (9.33) - Open loop not on C models
16-Sep-2010	Removed note about commands for LSPC.
26-Aug-2010	Added link to wide power sweep topic
11-Mar-2010	Added Use Last checkbox (A.09.20)
10-Feb-2009	MX New topic (A.08.50)

Sweep Settings

A sweep is a series of consecutive data point measurements taken over a specified sequence of stimulus values. You can make the following sweep settings:

- [Sweep Type](#)
 - [Linear / Log](#)
 - [Power Sweep](#)
 - [CW Time](#)
 - [Segment Sweep](#)
 - [Phase](#)
- [Sweep Time](#)
- [Sweep Setup](#)
 - [Stepped vs Analog](#)
 - [Fast Sweep](#)
 - [Dwell and Delay](#)
 - [Standard vs Point Sweep](#)

See [Triggering](#) and other 'Setup Measurements' topics

How to set Sweep Type

Using front-panel

HARDKEY [softkey] buttons

1. Press **SWEEP**
2. then [**Sweep Type**]

PNA Menu using a mouse

1. Click **Stimulus**
2. then **Sweep**
3. then **Sweep Type**

[Programming Commands](#)

Sweep Type dialog box help



Note: Sweep Settings are not applied until either **OK** or **Apply** is pressed.

Channel The active channel when Sweep Type was selected. Sweep settings will be applied to this channel.

Sweep Type

Linear Frequency Sets a linear frequency sweep that is displayed on a standard grid with ten equal horizontal divisions.

- **Start** Sets the beginning value of the frequency sweep.
- **Stop** Sets the end value of the frequency sweep.
- **Points** Sets the number of data points that the PNA measures during a sweep. Range: 2 to 20001. (Default is 201).

Log Frequency The source is stepped in logarithmic increments and the data is displayed on a logarithmic x-axis. This is usually slower than a continuous sweep with the same number of points.

- **Start** Sets the beginning value of the frequency sweep.
- **Stop** Sets the end value of the frequency sweep.
- **Points** Sets the number of data points that the PNA measures during a sweep. Range: 2 to 20001. (Default is 201).

Power Sweep Activates a power sweep at a single frequency that you specify. [Learn about power sweep](#)

- **Start** Sets the beginning value of the power sweep.
- **Stop** Sets the end value of the power sweep.
- **CW Frequency** Sets the single frequency where the PNA remains during the measurement sweep.

CW Time Sets the PNA to a single frequency, and the data is displayed versus time. [Learn more.](#)

- **CW Frequency** Sets the frequency where the PNA remains during the measurement.
- **Sweep Time** Sets the duration of the measurement, which is displayed on the X-axis.
- **Points** Sets the number of data points that the PNA measures during a sweep. Range: 2 to 20001.(Default is 201).

Segment Sweep Sets the PNA to sweep through user-defined sweep segments. [Learn how to make these settings.](#)

- **Independent Power Levels** Check to set the source power level for each segment. [Test port uncoupling](#) is also allowed.
- **Independent IF Bandwidth** Check to set the IF bandwidth for each segment.
- **Independent Sweep Time** Check to set the duration of the measurement for each segment.
- **X-Axis Point Spacing** Check to scale the X-Axis to include only the segments. [Learn more.](#)
- **Allow Arbitrary Segments** Check to allow arbitrary frequencies (overlapped or reverse sweeps). [Learn more](#)
- **Show Table** Shows the table that allows you to create and edit segments.
- **Hide Table** Hides the segment table from the screen.

Phase Sweep the phase of one or more sources relative to another source. Choose values between -360° and +360°. [Learn more.](#)

- **Start** Sets the beginning value of the phase sweep.
- **Stop** Sets the end value of the phase sweep.
- **CW Frequency** Sets the single frequency where the PNA remains during the sweep.

OK Applies setting changes and closes the dialog box.

Apply Applies setting changes and leaves the dialog box open to make more setting changes.

Cancel Closes the dialog. Setting changes that have been made since the last Apply button click are NOT applied.

Power Sweep

A power sweep either increases or decreases source power in discrete steps. Power sweep is used to characterize power-sensitive circuits, with measurements such as gain compression.

In the Sweep Type dialog, specify Start power, Stop power, and CW Frequency. Power can be swept over any attainable range within the [PNA ALC range](#).

Note: If the PNA has source attenuators, and the attenuation must be changed in order to achieve the requested start and stop power, click **Stimulus**, then **Power** then [Power and Attenuators](#).

The PNA does NOT allow a single power sweep over a range that requires attenuator switching. However, two power sweeps can be performed in different channels. The attenuators will not be allowed to switch continuously, but triggering can be performed using single or group triggering. [Learn more](#).

The remaining power settings apply in power sweep mode:

- Test Port Power setting is not available.
- Port Power can be coupled or uncoupled.
- Attenuator Control is always Manual.
- Power Slope (dB/GHz) is ignored (output frequency is CW).
- Click **Stimulus** then **Sweep**, then [Number of Points](#) to change the step size of the power sweep.

Notes:

- Using a [preference setting](#), you can specify whether to maintain source power at either the start power or stop power level at the end of a power sweep.
- Beginning with PNA Rev. A.08.50, Power Sweep has been optimized for speed. For highest measurement accuracy during a power sweep, it may be necessary to increase the [Dwell Time](#) to allow the source more time to settle.
- You may be able to perform a **60 dB power sweep** with Receiver Leveling. [Learn how](#).

Segment Sweep

Segment Sweep activates a sweep which consists of frequency sub-sweeps, called segments. For each segment you can define independent power levels, IF bandwidth, and sweep time.

Once a measurement calibration is performed on the entire sweep or across all segments, you can make calibrated measurements for one or more segments.

In segment sweep type, the analyzer does the following:

- Sorts all the defined segments in order of increasing frequency
- Measures each point
- Displays a single trace that is a composite of all data taken

Restrictions for segment sweep:

- The frequency range of a segment is not allowed to overlap the frequency range of any other segment.
- The number of segments is limited only by the combined number of data points for all segments in a sweep.
- The combined number of data points for all segments in a sweep cannot exceed 20001.
- All segments are FORCED to have power levels within the same attenuator range to avoid premature wear of the mechanical step attenuator. See [Power Level](#).

How to make segment sweep settings

**Using front-panel
HARDKEY [softkey] buttons**

1. Press **SWEEP**
2. then [**Sweep Type**]

PNA Menu using a mouse

1. Click **Stimulus**
2. then **Sweep**
3. then **Sweep Type**

Programming Commands

Insert Segment - adds a sweep segment before the selected segment. You can also click the "down" arrow on your keyboard to quickly add many segments.

Delete Segment - removes the selected segment.

Delete All Segments - removes all segments.

Note: At least ONE segment must be ON or [Sweep Type](#) is automatically set to **Linear**.

To Modify an Existing Segment

To make the following menu settings available, you must first show the segment table.

Click **View**, point to **Tables**, then click **Segment Table**.

	STATE	START	STOP	POI	IFBW	P1 PwR	P2 PwR	TIME
1	ON	20.000000 MHz	1.000000 GHz	21	10.0 kHz	17.00 dBm	0.00 dBm	2.474 msec
2	ON	1.000000 GHz	4.000000 GHz	21	35.0 kHz	17.00 dBm	0.00 dBm	630.000 µsec

The above graphic shows the Segment table with all independent settings selected, including source power uncoupled (two power settings).

STATE Click the box on the segment to be modified. Then use the up / down arrow to turn the segment ON or OFF.

START Sets start frequency for the segment. Click the box and type a value and the first letter of a suffix (**KHz**, **Mhz**, **GHz**). Or double-click the box to select a value.

STOP Sets stop frequency for the segment. Click the box and type a value and the first letter of a suffix (**KHz**, **Mhz**, **GHz**). Or double-click the box to select a value.

Note: The segment table truncates the frequency resolution. To verify the frequency resolution that you input, create a marker at the start or stop frequency settings.

POINTS Sets number of data points for this segment. Type a value or double-click the box to select a value.

To set IFBW, Power, and Sweep Time independently for each segment:

1. On the **Sweep** menu, click **Sweep Type**, then **Segment Sweep**.
2. Check the appropriate **Sweep Properties** boxes
3. Then click the box and type a value or double-click the box and select a value.

Note: If the following are NOT set, the entire sweep uses the channel IFBW, Power, and Time settings.

IFBW Sets the [IF Bandwidth](#) for the segment.

POWER Sets the [Power level](#) for the segment. You can also UNCOUPLE the test port power. See [Power Coupling](#).

TIME Sets the [Sweep time](#) for the segment.

X-Axis Point Spacing - Segment Sweep ONLY

This feature affects how a segment trace is drawn on the screen.

How to select X-Axis Point Spacing

On the [Sweep Type](#) dialog box, click **Segment Sweep**

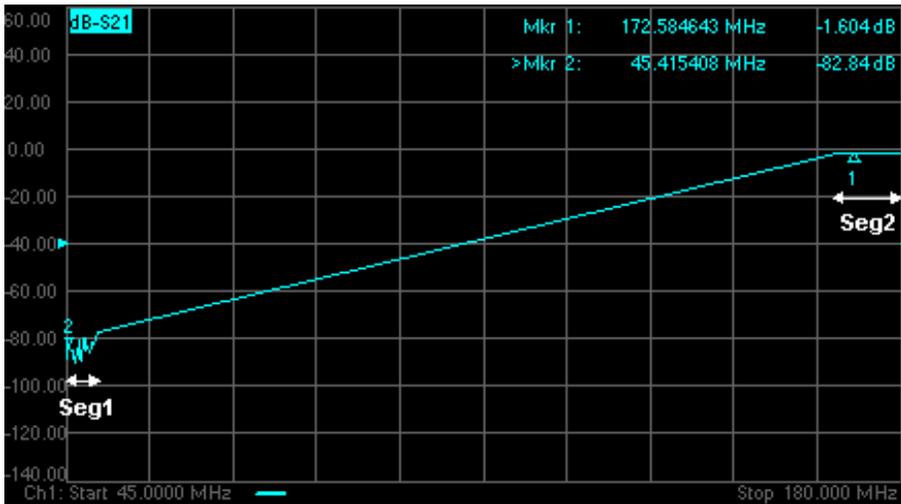
Then check **X-Axis Point Spacing**

- **Without X-axis point spacing**, a multi-segment sweep trace can sometimes result in squeezing many measurement points into a narrow portion of the x-axis.

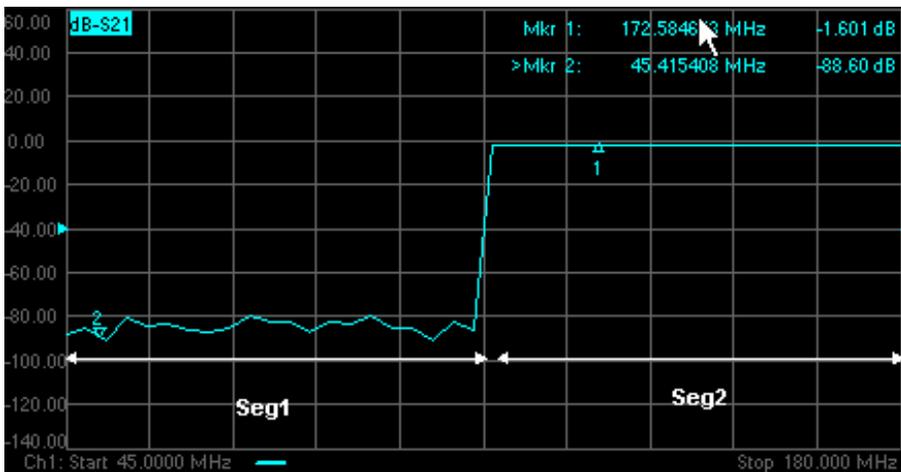
- **With X-axis point spacing**, the x-axis position of each point is chosen so that all measurement points are evenly spaced along the x-axis.

For example, given the following two segments:

	STATE	START	STOP	POINTS
1	ON	45.000000 MHz	50.000000 MHz	21
2	ON	170.000000 MHz	180.000000 MHz	21



Without X-Axis Point Spacing



With X-Axis Point Spacing

Arbitrary Segment Sweep

This feature allows arbitrary frequencies to be entered into the segment sweep table. With this capability, segments can have:

- overlapping frequencies.
- the stop frequency less than the start frequency (reverse sweep).

How to enable Arbitrary Segment Sweep

1. On the [Sweep Type](#) dialog box, click **Segment Sweep**
2. Check **Allow Arbitrary Segment Sweep**

Notes:

- Unusual results may occur when using arbitrary sweep segments with markers, display settings, limit lines, formatting, and some calibration features.
- When Allow Arbitrary Segment is checked, [X-axis point spacing](#) is automatically turned ON.

Sweep Time

The PNA automatically maintains the fastest sweep time possible with the selected measurement settings. However, you can increase the sweep time to perform a slower sweep.

How to set Sweep Time

Using front-panel HARDKEY [softkey] buttons

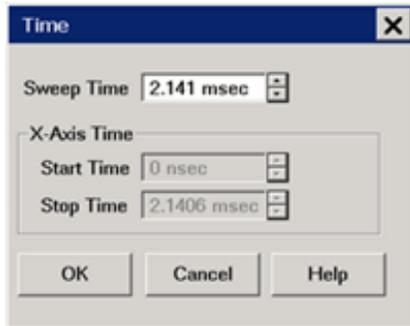
1. Press **SWEEP**
2. then **[Time]**

PNA Menu using a mouse

1. Click **Stimulus**
2. then **Sweep**
3. then **Sweep Time**

Programming Commands

Time dialog box help



Sweep Time Specifies the time the PNA takes to acquire data for a sweep. The maximum sweep time of the PNA is 86400 seconds or 1 day. [Learn about other settings that affect sweep speed.](#)

Note: If sweep time accuracy is critical, use ONLY the up and down arrows next to the sweep time entry box to select a value that has been calculated by the PNA. Do NOT type a sweep time value as it will probably be rounded up to the closest calculated value. This rounded value will not be updated in the dialog box.

X-Axis Time Set Start and Stop time to be displayed on the X-axis. These settings are ONLY available for [Pulse Profile](#) measurements.

- The actual sweep time includes this acquisition time plus some "overhead" time.
- The PNA automatically maintains the fastest sweep time possible with the selected measurement settings. However, you can increase the sweep time using this setting.
- Enter **0** seconds to return the analyzer to the fastest possible sweep time.
- The Sweep Time setting is applied to the active channel.
- The sweep time is per sweep. A full 2-port cal requires two sweeps, both using the specified sweep time. [Learn more.](#)
- A **Sweep Indicator**  appears on the data trace when the Sweep Time is 0.3 seconds or greater, or if trigger is set to [Point](#). The indicator is located on the last data point that was measured by the receiver. If the indicator is stopped (point sweep mode) the source has already stepped to the next data point.

Sweep Setup

How to make Sweep Setup settings

Using front-panel

HARDKEY [softkey] buttons

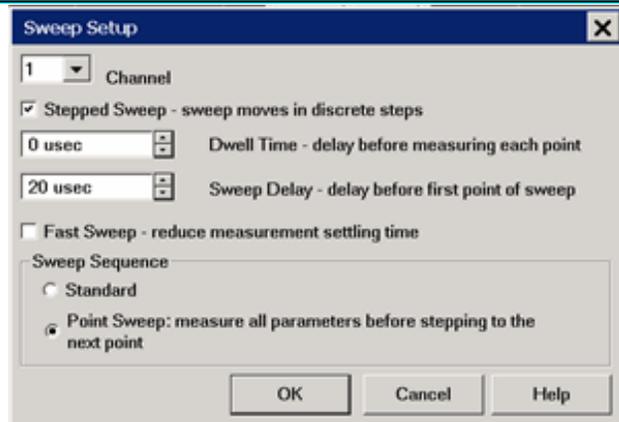
1. Press **STIMULUS**
2. then [**Sweep**]
3. then [**Sweep Setup**]

PNA Menu using a mouse

1. Click **Stimulus**
2. then **Sweep**
3. then **Sweep Setup**

Programming Commands

Sweep Setup dialog box help



Channel Specifies the channel that the settings apply to.

Stepped Sweep When checked (Stepped Sweep) the PNA source is tuned, then waits the specified Dwell time, then takes response data, then tunes the source to the next frequency point. This is slower than Analog Sweep, but is more accurate when testing electrically-long devices.

When cleared (Analog Sweep) the PNA takes response data AS the source is sweeping. The sweep time is faster than Stepped, but could cause measurement errors when testing electrically-long devices.

When the dialog checkbox is cleared, the PNA could be in either Analog or Step mode. The mode can change from sweep to sweep. There is **NO way** to determine whether the PNA is in Analog or Stepped Sweep. If you want to be sure what the current sweep mode is, then switch it to Stepped.

Stepped sweep is automatically selected for a number of reasons. Here are some of the reasons:

- [IF Bandwidth](#) is at, or below, 1 kHz.
- [Source Power Correction](#) is ON unless doing CW measurement.
- When more than one source is turned ON ([multisource PNA models](#)).
- When step mode is a faster way to take the data.
- For all [FOM](#) and [FCA](#) measurements.
- For all [ADC measurements](#).
- For all [MMwave measurements](#).

Dwell Time Specifies the time the source stays at each measurement point before the analyzer takes the data. Only applies to stepped sweep. The maximum dwell time is 100 seconds. See also [Electrically Long Devices](#).

Sweep Delay Specifies the time to wait just before acquisition begins for each sweep. This delay is in addition to Dwell Time (per point) and [External Trigger delay](#) if enabled.

Fast Sweep NOT available on [N5264A](#) and [N523xA](#) models. When checked, in Analog Sweep mode the PNA source settling times are shortened in both frequency and power-control (ALC) circuits. In Stepped Sweep mode, the settling time at ALL data points are shortened. This nearly doubles the sweep speed at preset conditions, but at the expense of frequency accuracy and a few dB of amplitude variation. For ratioed measurements, such as S-Parameters, these errors substantially ratio out.

- By default, Fast Sweep is always OFF to provide maximum accuracy and stability.
- Fast Sweep is NOT allowed with [Power Limit](#) enabled.
- **Note:** PNA performance specifications do NOT apply in Fast Sweep.

Alternate Sweeps This selection is no longer available from the user interface. The inherent [crosstalk](#) on the PNA is sufficient without this feature. Remotely, this setting is still available and sweeps can be performed alternately. However, receivers can NOT be turned off individually on the PNA-X, N522xA, or N523xA models.

Sweep Sequence

Standard Sweep When checked, the PNA sweeps all data points for each source port in turn. For a 2-port PNA, this means that all data points are swept in the forward direction, then all data points are swept in the reverse direction. Even when NO reverse parameters are displayed (S22 or S12), reverse measurements are necessary when a full 2-port calibration is correcting the channel. This is the default behavior. [Learn more.](#)

Point Sweep Available ONLY on standard S-parameter channels. When checked, the PNA measures all parameters at each frequency point before stepping to the next frequency. The display trace is updated as each data point is measured.

- Point sweep usually results in slower sweeps and is useful only in rare circumstances.
- Point sweep is the same as stepped sweep mode on the 8510 and 8530.

Last modified:

25-Jun-2012	Point sweep edit (JE)
3-May-2012	Edited for N523x models
6-Feb-2012	Edited Alternate sweep
27-Oct-2010	Added Phase Sweep and Time diag (9.30)
26-Aug-2010	Added link to Wide Power Sweep
1-Mar-2010	Sweep Delay and Point sweep now on C models
28-Sep-2009	Added segment frequency resolution note.
6-Apr-2009	Replaced N5242A with PNA-X
5-Mar-2009	Added faster power sweep
17-Feb-2009	Added point sweep availability
26-Jan-2009	Removed 'band-crossings' at fast sweep.
10-Oct-2008	Added point sweep (8.35)
3-Sep-2008	Removed legacy content
19-Aug-2008	Added Fast Sweep
6-Jun-2008	Added uncoupled power sweep
15-Apr-2008	Updated again for Step mode
14-Nov-2007	Added Alternate Sweep note
21-Jun-2007	Increased max data points
3-May-2007	Updated Step mode conditions

Trigger

A trigger is a signal that causes the PNA to make a measurement sweep. The PNA offers great flexibility in configuring the trigger function.

View the interactive [Trigger Model](#) animation to see how triggering works in the PNA.

- [How to Set Trigger](#)
- [Source](#)
- [Scope](#)
- [Channel Settings](#)
- [Restart](#)
- [External and Auxiliary Triggering](#) (separate topic)

[See other 'Setup Measurements' topics](#)

How to set Triggering

Using front-panel HARDKEY [softkey] buttons

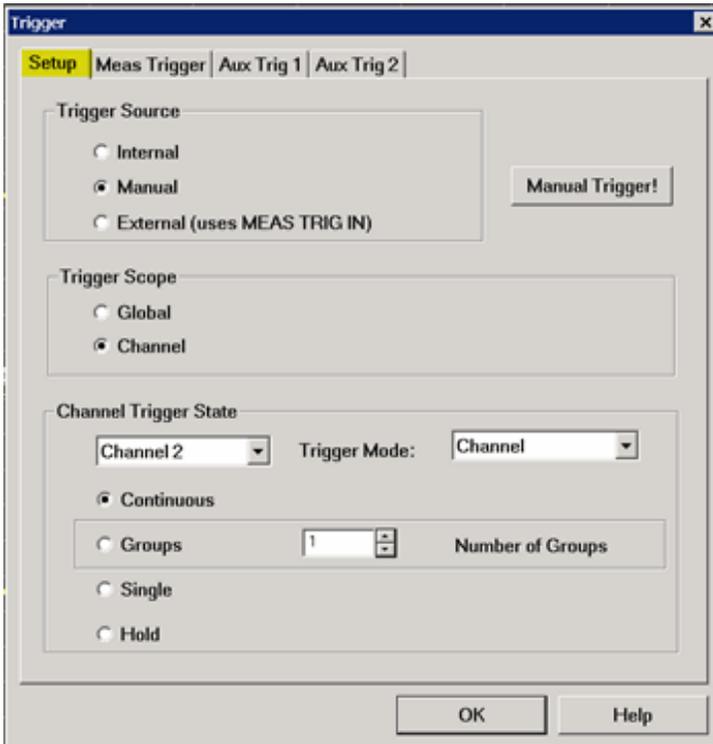
1. Press TRIGGER
2. then [Trigger...]

PNA Menu using a mouse

1. Click **Stimulus**
2. then **Trigger**
3. then **Trigger**

[Programming Commands](#)

Note: The **Continuous**, **Single**, and **Hold** settings apply ONLY to the active channel. These settings are available from the Trigger menu, Active Entry keys, and softkeys



Trigger Setup dialog box help

View the interactive [Trigger Model](#) animation to see how triggering works in the PNA.

Trigger Source

These settings determine **where** the trigger signals originate for all existing channels. A valid trigger signal can be generated only when the PNA is not sweeping.

Internal Continuous trigger signals are sent by the PNA as soon as the previous measurement is complete.

Manual One trigger signal is sent when invoked by the Trigger button, the active toolbar, or a programming command.

External Trigger signals sent out or received from various connectors on the rear panel. [Learn more about External and AUX Triggering.](#)

Manual Trigger! - Manually sends one trigger signal to the PNA. Available ONLY when Manual trigger is selected.

Trigger Scope

These settings determine **what** is triggered.

Global All channels not in Hold receive the trigger signal [Default setting]

Channel Only the next channel that is not in Hold receives the trigger signal. This is not obvious or useful unless Trigger Source is set to Manual. This setting enables [Point Sweep](#) mode.

Channel Trigger State

These settings determine **how many** trigger signals the channel will accept.

Continuous The channel accepts an infinite number of trigger signals.

Groups The channel accepts only the number of trigger signals that is specified in the Number of Groups text box, then goes into Hold. Before selecting groups you must first increment the Number of Groups text box to greater than one.

Number of Groups Specify the number of triggers the channel accepts before going into Hold. If in Point Sweep, an entire sweep is considered one group.

First increment to desired number, then select 'Groups'.

Single The channel accepts ONE trigger signal, then goes into Hold.

Another way to trigger a single measurement is to set [Trigger Source](#) to Manual, then send a **Manual trigger**. However, ALL channels are single triggered.

Hold The channel accepts NO trigger signals.

Trigger Mode

These settings determine what EACH signal will trigger.

Sweep and **Point** modes are available ONLY when both [Trigger Source](#) = MANUAL or EXTERNAL AND [Trigger Scope](#) = CHANNEL.

- **Channel** Each trigger signal causes **ALL traces** in that channel to be swept in the order specified below.
- **Point** Each Manual or External trigger signal causes one data point to be measured. Subsequent triggers go to the same trace until it is complete, then other traces in the same channel are swept in the order specified below. When in Groups or Single trigger, the count is decremented by one after ALL data points on ALL traces in the channel are measured. See Also, the (point) [Sweep Indicator](#) and [SCPI Triggering example](#) for use with External.
- **Trace** Available ONLY when [Point Sweep](#) is selected. Each trigger signal causes two identical measurements to be triggered separately - one trigger signal is required for each measurement. Other trigger mode settings cause two identical parameters to be measured simultaneously. Trace triggering is NOT permitted when a channel is using a 2 port (or

more) S-Parameter calibration.

- **Sweep** Each Manual or External trigger signal causes **ALL traces that share a source port** to be swept in the order specified below. When in Groups or Single trigger, the count is decremented by one after ALL traces in ALL directions are swept.

When multipoint correction is ON, which requires sweeps in more than one direction, traces on the screen will not update until all of the relevant directions have been swept. For example, with all four 2-port S-Parameters displayed:

- When Full 2-port correction is ON, trigger 1 causes NO traces to update; trigger 2 causes ALL S-Parameters to update. [Learn more about sweeps with correction ON.](#)
- When correction is OFF, trigger 1 causes S11 and S21 to update; trigger 2 causes S22 and S12 to update.

Trace Sweep Order

For ALL Trigger Modes, trigger signals continue in the same channel until all traces in that channel are complete. Triggering then continues to the next channel that is not in HOLD.

Traces within each channel are always swept in the following order:

- Traces are swept sequentially in source-port order. For example, in a channel with all four 2-port S-parameters, first the source port 1 traces (S11 and S21) are swept simultaneously. Then the source port 2 traces (S22 and S12) are swept simultaneously.
- In addition, when [Alternate sweep](#) is selected, traces are swept sequentially in source-port / receiver-port order. In the above example, first the S11 trace is swept, then S21, then S12, then S22.

Restart (Available only from the Trigger menu) Channels in Hold are set to single trigger (the channel accepts a single trigger signal). All other settings are unaffected, including decrementing trigger Groups.

See Also

- [External and AUX Triggering.](#)
- Interactive [Trigger Model](#) animation

Last modified:

10-Oct-2008	Added Trace trigger (8.35)
3-Sep-2008	Removed legacy content
26-Oct-2007	Added Trigger Mode
15-Dec-2006	Added MX capability
9/12/06	Added link to programming commands

External and Auxiliary Triggering

External and Auxiliary triggering are both used to synchronize the triggering of the PNA with other equipment.

Note: When an External Source is configured as an External Device, the PNA automatically controls all trigger settings. Do NOT make additional trigger settings. [Learn more.](#)

- [Overview](#)
- How to make Trigger Settings:
 - [Auxiliary Triggering](#)
 - [Meas Trig \(IN\) Dialog](#)
 - [Pulse Triggering](#)

See Also

- [Controlling a Handler](#)
 - [Synchronizing an External Source](#)
 - [PNA Triggering](#)
 - [Pulse Triggering](#) (separate topic)
-

Overview

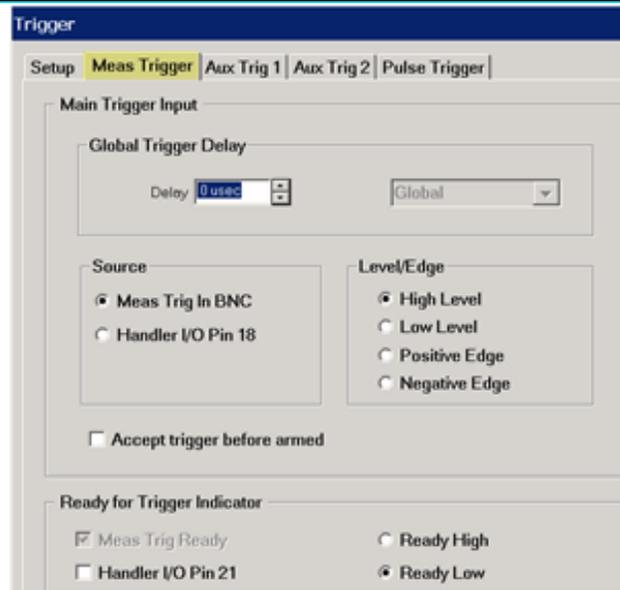
Ready Signals versus Trigger Signals

A 'Ready for Trigger' signal is different from a Trigger signal. The ready signal indicates that the instrument sending the signal is ready for measurement. The instrument receiving the ready signal would then send a trigger signal, indicating that the measurement will be, or has been, made. Usually the slower instrument sends the trigger signal.

Learn more about each type of triggering signal:

- [Meas Trig RDY and Meas Trig IN](#) - This pair of signals is easy to use and limited in ability to configure.
- [AUX TRIG OUT and AUX TRIG IN](#) - These two pair of connectors and signals are highly configurable. Use them to synchronize with any number of devices and equipment.

Meas (External) Trigger dialog box help



[See how to access the Trigger Dialog](#)

Meas Trig RDY and Meas Trig IN

The [MEAS TRIG](#) connectors are located on the PNA rear-panel.

These signals can be used when the PNA is communicating with a slow mechanical device. A material handler is very mechanical and takes a relatively long time to load and discharge parts. Here is how these signals work together to communicate:

1. The PNA sends a 'Ready' signal when it is ready to make a measurement.
2. The external device sends a trigger signal to the PNA when it is ready for a measurement.
3. Additional signals are available on the PNA Handler I/O to indicate that the PNA sweep has ended, and that the handler can setup for the next measurement. See [Material Handler I/O description](#).

Dialog Settings

To cause the PNA to respond to Meas Trig IN or Handler I/O signals, select **External** on the [Trigger Setup tab, Source](#) setting.

Also on the Trigger Setup tab, Scope setting, choose whether one external trigger signal will apply to ALL channels (Global) or one trigger signal per Channel. The following settings apply accordingly.

Main Trigger Input

Global / Channel Trigger Delay After an external trigger is received, the start of the sweep is held off for this specified amount of time plus any inherent latency.

- When [Trigger Scope](#) = Channel, the delay value is applied to the specified channel.
- When Trigger Scope = Global, the same delay value is applied to ALL channels.

Source The PNA accepts Trigger IN signals through the following rear-panel connectors:

- [Meas Trig IN BNC](#)
- [Handler I/O Pin 18](#)

Level / Edge

High Level The PNA is triggered when it is armed (ready for trigger) and the TTL signal at the select input is HIGH.

Low Level The PNA is triggered when it is armed (ready for trigger) and the TTL signal at the select input is LOW.

Positive Edge After the PNA arms, it will trigger on the next positive edge. If [Accept Trigger Before Armed](#) is set, PNA will trigger as soon as it arms if a positive edge was received since the last data was taken.

Negative Edge After the PNA arms, it will trigger on the next negative edge. If [Accept Trigger Before Armed](#) is set, PNA will trigger as soon as it arms if a negative edge was received since the last data was taken.

Accept Trigger Before Armed When checked, as the PNA becomes armed (ready to be triggered), the PNA will immediately trigger if any triggers were received since the last taking of data. The PNA remembers only one trigger signal. All others are ignored.

- When this checkbox is cleared, any trigger signal received before PNA is armed is ignored.
- This feature is only available when positive or negative EDGE triggering is selected.
- Configure this setting remotely using [CONTrol:SIGNal](#) (SCPI) or [ExternalTriggerConnectionBehavior](#) (COM).

Ready for Trigger Indicator (OUT)

On the PNA, when External is selected on the Trigger Setup tab, then both Meas Trig IN and Meas Trig Ready (OUT) are enabled.

Choose a connector to send the PNA Ready OUT signal:

- [Meas Trig RDY](#)
- [Handler I/O p21](#)

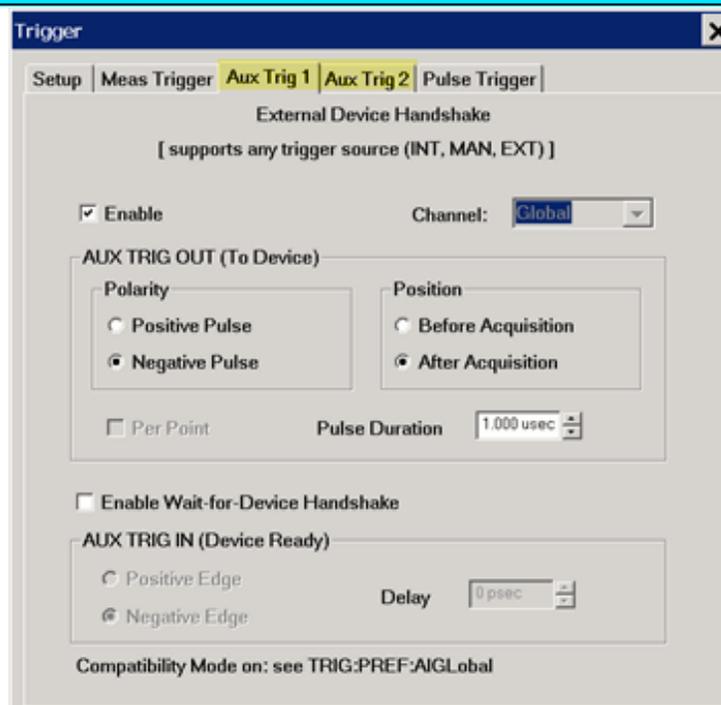
Choose Polarity of the 'Ready OUT' signal.

- **Ready High** - TTL High indicates the PNA is ready for trigger
- **Ready Low** - TTL Low indicates the PNA is ready for trigger (default setting).

See Also

- [Pulse Trigger Tab](#)
- [Learn how to External Trigger during Calibration](#)

Aux Trig 1 - Aux Trig 2 dialog box help



[See how to access the Trigger Dialog](#)

Note: When an External Source is configured as an External Device, the PNA automatically controls all trigger settings. Do NOT make additional trigger settings. [Learn more.](#)

AUX TRIG OUT and AUX TRIG IN

See the AUX TRIG (1&2) connectors on the [PNA rear-panel](#).

These signals are highly configurable. They can be used with all types of external devices to send and receive signals. However, it is important to note that either Aux Trig INPUT does NOT trigger the PNA. That signal must be selected. See step 2 in the following procedure.

1. An external source sends a 'Ready' signal to the PNA (at the Aux Trig IN connector) when it is settled at a frequency.
2. After receiving the Ready signal, the PNA begins the measurement when it receives a Trigger signal from the specified [Trigger Source](#):
 - **Internal** - Measurement begins immediately.
 - **Manual** - Measurement begins when the PNA Trigger button is pressed.
 - **External** - Measurement begins when [Meas Trig In](#) signal is received from an external device. This must be configured independently.
3. The Aux Trig OUT signal can be configured to be sent either just BEFORE the measurement is made or AFTER the measurement is complete. When communicating ONLY with an external source, the Aux Trig OUT signal should be sent AFTER the measurement is complete to indicate that the external source can setup for the next measurement.

Dialog Settings

The Aux Trig 1 and Aux Trig 2 tabs are identical. Two pair of connectors are available to allow two external devices to be controlled simultaneously.

Enable Check to use the Aux1 or Aux2 connectors to output signals to an external device.

Channel: This setting is controlled by a [PNA Preference setting](#).

- **Global** - ALL Aux Trig settings apply to ALL channels. The Per Point setting (see below) is made on the [Trigger Setup tab](#) which also applies to ALL channels.
- **Channel** - ALL Aux Trig settings apply to the specified channel. Each channel can be configured independently.

AUX TRIG OUT (To Device)

The following settings control the properties of the signals sent out the rear panel [AUX TRIG OUT \(1&2\) connectors](#):

Polarity

Positive Pulse Outgoing pulse is positive.

Negative Pulse Outgoing pulse is negative.

Position

Before Acquisition Pulse is sent immediately **before** data acquisition begins.

After Acquisition Pulse is sent immediately **after** data acquisition is complete.

Per Point Check to cause a trigger output to be sent for each data point. Clear to send a trigger output for each sweep.

When the Aux Trig - "[Global](#)" [PNA Preference](#) is selected, then the Point setting is made on the [Trigger Setup tab](#). It then applies to ALL channels. When more than one channel is present, the channel setting that was made last is used.

Pulse Duration Specifies the duration of the positive or negative output trigger pulse.

Enable Wait-for-Device Handshake

When checked, the PNA waits indefinitely for the input line at the rear panel [AUX TRIG IN \(1&2\) connectors](#) to change to the specified level before acquiring data. This signal indicates that the external device is ready for PNA data acquisition. If the signal arrives before the PNA is ready to acquire data, it is latched (remembered).

When NOT checked, the PNA does not wait, but outputs trigger signals when the PNA is ready.

This signal does NOT trigger the PNA. The trigger signal is generated from [Trigger Sources](#): Internal, Manual, or External.

IN (READY)

Positive Edge PNA responds to the leading edge of a pulse on the Aux1 or Aux2 In connector.

Negative Edge PNA responds to the trailing edge of a pulse on the Aux1 or Aux2 In connector.

Delay Time that the PNA waits after receiving the Handshake input before data acquisition begins.

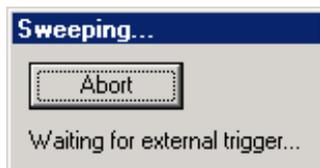
See Also

- See how to use these connectors to [synchronize with External Sources](#).
- [Pulse Triggering](#)

Note: Beginning with PNA Rev 6.0, Guided and Unguided Calibration CAN be performed in External Trigger mode. With this optional behavior, while Trigger Source is set to External, trigger signals must be sent for Calibration sweeps. This behavior does not apply to FCA calibrations.

You can [set a Preference](#) to revert to pre-6.0 behavior - the PNA calibrates using Internal trigger signals while Trigger Source is set to External.

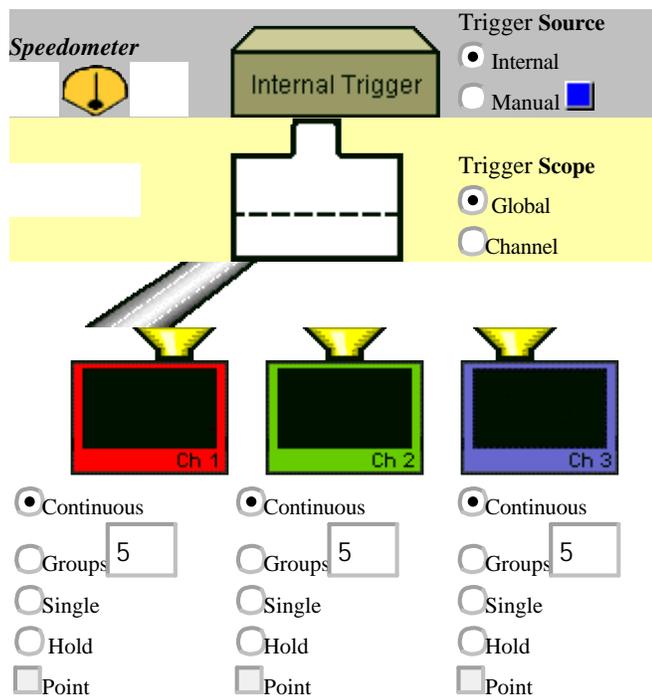
The following dialog box appears on the PNA screen while the PNA is waiting for an External trigger signal.



Click **Abort** to cancel the wait for a trigger signal.

Last Modified:

11-Apr-2012	Remove old content
9-Sep-2011	Major re-write
3-Aug-2011	Clarification for Pulse
15-Mar-2010	Added Pulse Trig links
13-Aug-2009	Corrected Aux Trig IN link
22-Apr-2008	Added link to UI preference
14-Mar-2008	Added Ready Out polarity
3-Mar-2008	Many edits
25-Jan-2007	MX New topic



[About the trigger model](#)

Read [Text description](#) of triggering behaviors.

This model does not include the new [Sweep trigger mode](#).

Data Format

A data format is the way the PNA presents measurement data graphically. Pick a data format appropriate to the information you want to learn about the test device.

- [How to set Format](#)
- [Rectangular \(Cartesian\) Display Formats](#)
- [Polar](#)
- [Smith Chart](#)

[See other 'Setup Measurements' topics](#)

How to set the Display Format

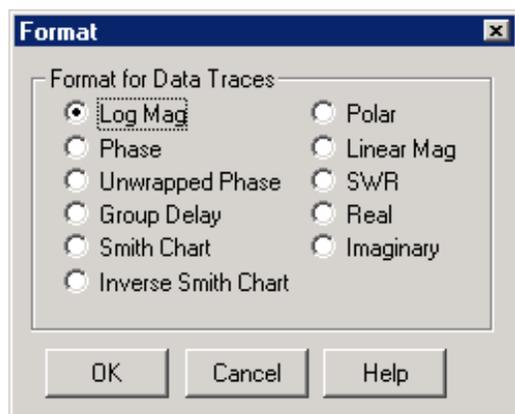
Using front-panel HARDKEY [softkey] buttons

1. Press **FORMAT**

PNA Menu using a mouse

1. Click **Response**
2. then **Format**

[Programming Commands](#)



Format dialog box help

- [Log Mag](#)
- [Phase / Unwrapped Phase](#)
- [Group Delay](#)
- [Smith / Inverse Smith Chart](#)
- [Polar](#)
- [Linear Mag](#)
- [SWR](#)
- [Real](#)
- [Imaginary](#)

Rectangular Display Formats

Seven of the nine available data formats use a rectangular display to present measurement data. This display is also known as Cartesian, X/Y, or rectilinear. The rectangular display is especially useful for clearly displaying frequency response information of your test device.

- Stimulus data (frequency, power, or time) appears on the X-axis, scaled linearly
- Measured data appears on the Y-Axis.

Log Mag (Logarithmic Magnitude) Format

- Displays Magnitude (no phase)
- Y-axis: dB
- Typical measurements:
 - Return Loss
 - Insertion Loss or Gain

Phase Format

Measures the phase of a signal relative to the calibration reference plane with a range of +/- 180 degrees.

- Displays Phase (no magnitude)
- Y-axis: Phase (degrees)
- The trace 'wraps' every 180 degrees for easier scaling.
- Typical Measurements:
 - [Deviation from Linear Phase](#)

Unwrapped Phase

- Same as Phase, but without 180 degree wrapping.

Note: Phase is unwrapped by comparing the phase from one data point to the next. If the phase difference between two points is greater than 180 degrees, or if the phase of the first data point is greater than 180 degrees from DC, then the phase measurement is probably NOT accurate.

Group Delay Format

- Displays signal transmission (propagation) time through a device
- Y-axis: Time (seconds)
- Typical Measurements:
 - Group Delay

See Also:

[Group Delay](#) (Measurement)

[Comparing the PNA Delay Functions.](#)

[Phase Measurement Accuracy](#)

Linear Magnitude Format

- Displays positive values only
- Y-axis: Unitless (**U**) for ratioed measurements
Watts (**W**) for unratioed measurements.
- Typical Measurements:
 - reflection and transmission coefficients (magnitude)
 - time domain transfer

SWR Format

- Displays reflection measurement data calculated from the formula $(1+\rho)/(1-\rho)$ where ρ is reflection coefficient.
- Valid only for reflection measurements.
- Y axis: Unitless
- Typical Measurements:
 - SWR

Real Format

- Displays only the real (resistive) portion of the measured complex data.
- Can show both positive and negative values.
- Y axis: Unitless
- Typical Measurements:
 - time domain
 - auxiliary input voltage signal for service purposes

Imaginary Format

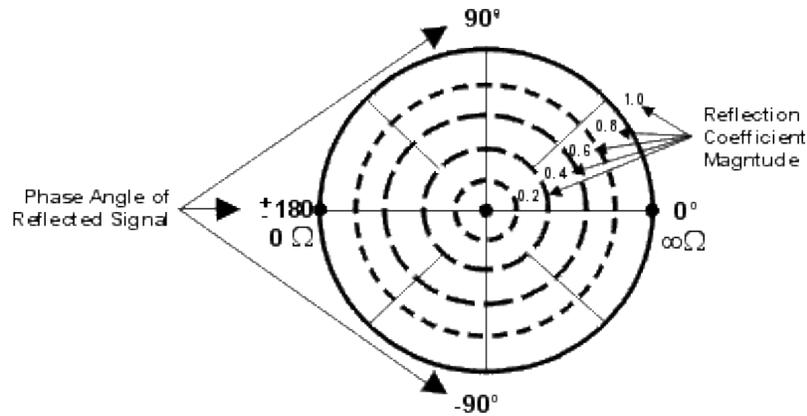
- Displays only the imaginary (reactive) portion of the measured data.
- Y - axis: Unitless
- Typical Measurements:
 - impedance for designing matching network

Polar Format

Polar format is used to view the magnitude and **phase** of the reflection coefficient (Γ) from your S_{11} or S_{22} measurement.

You can use Markers to display the following:

- Linear magnitude (in units) or log magnitude (in dB)
- Phase (in degrees)



- The dashed circles represent reflection coefficient. The outermost circle represents a reflection coefficient (Γ) of 1, or total reflected signal. The center of the circle represents a reflection coefficient (Γ) of 0, or no reflected signal.
- The radial lines show the phase angle of reflected signal. The right-most position corresponds to zero phase angle, (that is, the reflected signal is at the same phase as the incident signal). Phase differences of 90° , $\pm 180^\circ$, and -90° correspond to the top, left-most, and bottom positions on the polar display, respectively.

Smith Chart Format

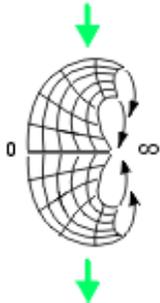
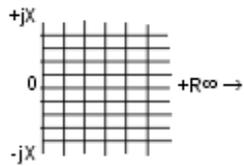
The Smith chart is a tool that maps the complex reflection coefficient (Γ) to the test device's impedance.

In a Smith chart, the rectilinear impedance plane is reshaped to form a circular grid, from which the series resistance and reactance can be read ($R + jX$).

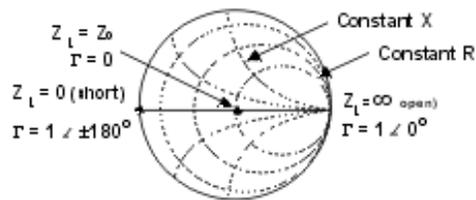
You can use Markers to display the following:

- Resistance (in units of ohms)
- Reactance as an equivalent capacitance (in units of farads) or inductance (in units of henrys)

Rectilinear Impedance Plane



Smith Chart

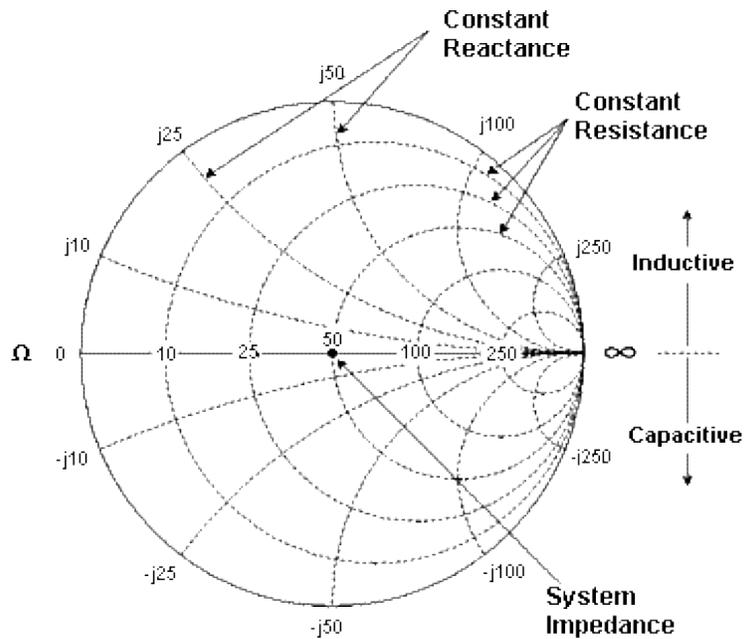


Inverse Smith Chart (also known as Admittance)

Same as standard Smith Chart , except:

- The plot graticule is reversed right-to-left.
- Admittance (in units of siemens) instead of resistance.

Interpreting the Smith Chart



- Every point on the Smith Chart represents a complex impedance made up of a real resistance (r) and an imaginary reactance ($r+jX$)
- The horizontal axis (the solid line) is the real portion of the impedance - the resistance. The center of the horizontal axis always represents the system impedance. To the far right, the value is infinite ohms (open). To the far left, the value is zero ohms (short)
- The dashed circles that intersect the horizontal axis represent constant resistance.
- The dashed arcs that are tangent to the horizontal axis represent constant reactance.
- The upper half of the Smith chart is the area where the reactive component is positive and therefore inductive.
- The lower half is the area where the reactive component is negative and therefore capacitive.

Last modified:

- 4-Jan-2010 Added link to group delay measurement
- 5-Aug-2009 Removed scale
- 6-Oct-2008 Added unwrapped phase note
- 3-Sep-2008 Removed legacy content
- 9/12/06 Added link to programming commands
- 9/27/06 MX Added UI

Scale

The Scale, Reference Level and Reference Position settings (along with [Format](#)) determine how the data trace appears on the PNA screen.

- [Scale, Reference Level and Position](#)
- [Scale Coupling](#)
- [Magnitude Offset](#)

[See other 'Setup Measurements' topics](#)

Scale, Reference Level and Position

The Scale, Reference Level and Reference Position settings (along with format) determine how the data trace appears on the PNA screen.

How to set Scale, Reference Level, and Position

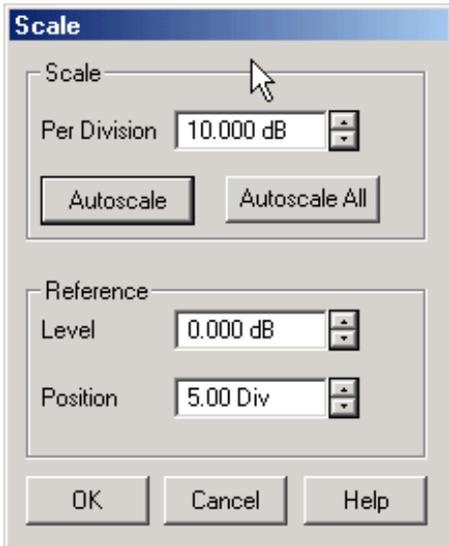
Using front-panel HARDKEY [softkey] buttons

1. Press **SCALE**

PNA Menu using a mouse

1. Click **Response**
2. then **Scale**
3. then **Scale**

[Programming Commands](#)



Scale dialog box help

Note: Beginning with PNA Rev. 9.0, the scale settings are set to couple with other traces in each window. The following settings assume that Scale Coupling is set to OFF. [Learn more about Scale Coupling.](#)

Scale

Per Division Sets the value of the vertical divisions of a rectangular display format. In Polar and Smith Chart formats, scale sets the value of the outer circumference. Range: 0.001dB/div to 500 dB/div.

Tip: Click on the Y-axis labels, then use a mouse scroll wheel to change scale in preset increments. Or Right-click on **Y-axis** annotation to change Scale.

Autoscale - Automatically sets value of the vertical divisions and reference value to fit the ACTIVE data trace within the grid area of the screen. The stimulus values and reference position are not affected.

The analyzer determines the smallest possible scale factor that will allow all the displayed data to fit onto 80 percent of the vertical grid.

The reference value is chosen to center the trace on the screen.

Tip: Double click on the Y-axis labels to autoscale the active trace.

Autoscale All Automatically scales ALL data traces in the ACTIVE WINDOW to fit vertically within the grid area of the screen.

Reference

Level In rectangular formats, sets the value of the reference line, denoted by  on the PNA screen. Range: -500 dB to 500 dB.

In Polar and Smith chart formats, reference level is not applicable.

Tip: Click on the Y-axis labels, then drag up or down to change the reference level in preset increments.

Position In rectangular formats, sets the position of the reference line. Zero is the bottom line of the screen and ten is the top line. Default position is five (middle).

In Polar and Smith chart formats, reference position is not applicable.

Tip: Click on the triangle , then drag up or down to change the reference position in preset increments.

Scale Coupling

With Scale Coupling enabled, traces that have the same format will have the same Scale, Reference Level, and Reference Position. You can choose to couple the scale of traces that are in the same window, couple the scale of all traces in all windows, or to have NO coupling.

How to set Scale Coupling

1. Right-click on the Y-axis labels of a window
2. then select **Scale Coupling**

OR

Using front-panel HARDKEY [softkey] buttons

1. Press **SCALE**
2. then **More**
3. then **Scale Coupling**

PNA Menu using a mouse

1. Click **Response**
2. then **Scale**
3. then **Scale Coupling**

 **Programming Commands** 



Scale Coupling dialog box help

Allows traces that share the same [format](#) to have the same [Scale](#), [Reference Level](#) and [Reference Position](#).

Coupling Method

Off - No coupling. Traces are scaled individually. Default setting.

Window - All traces with the same format in each selected window share the same scale settings.

All - All traces in ALL selected windows with the same format share the same scale settings.

- When **Window** or **All** coupling is enabled, the scale settings for the active trace are assumed by other coupled traces with the same format.
- When there are traces with a different format present, all traces with that format assume the trace settings of the lowest-numbered trace of that format.
- Once enabled, scale settings for all coupled traces with the same format can be changed with any coupled trace being active.

Selected Windows

Available when either the **Window** or **All** method is selected. Selected windows will participate in scale coupling. All windows are selected by default. Clear a checkbox to 'Opt-out' of scale coupling for that window.

About Autoscale and Scale Coupling

Autoscale (not Autoscale All) affects the active trace in the active window. All traces that are coupled to this trace assume the new scale settings of the active trace. This could cause some traces to NOT show on the screen.

Autoscale All with Coupling Method...

- **Off** - All traces in the active window are autoscaled independently.
- **Window** - All traces in each selected window are autoscaled to fit within a common set of scaling factors.
- **All** - All traces in all selected windows are autoscaled to fit within a common set of scaling factors.

Magnitude Offset

Magnitude Offset allows you to offset the magnitude (not phase) data by a fixed and / or sloped value in dB. If the display format is Linear Magnitude or Real (unitless), the conversion from dB is performed and the correct amount of offset is implemented.

How to set Magnitude Offset

Using front-panel

HARDKEY [softkey] buttons

1. Press **SCALE**
2. then **[More]**
3. then **[Magnitude Offset]**

PNA Menu using a mouse

1. Click **Response**
2. then **Scale**
3. then **Magnitude Offset**

Programming Commands



Magnitude Offset dialog box help

The Magnitude offset setting affects only the active trace.

Offset Offsets the entire data trace by the specified value.

Slope Offsets the data trace by a value that changes with frequency. The offset slope begins at 0 Hz.

For your convenience, the offset value at the start frequency is calculated and displayed.

See where this operation is performed in the [data processing chain](#).

Last modified:

1-Dec-2011 Added tips

Pre-configured Measurement Setups

- [Pre-configured setups for NEW measurements](#)
- [Pre-configured arrangements for EXISTING measurements](#)

Before reading this topic, it is important to understand [Traces, Channels, and Windows](#) in the PNA.

[See other 'Setup Measurements' topics](#)

Pre-configured Setups for NEW Measurements

Each of the following setups **creates new traces**. Existing traces and their settings will be lost, unless you first save them.

How to select a pre-configured measurement setup

Using front-panel HARDKEY [softkey] buttons

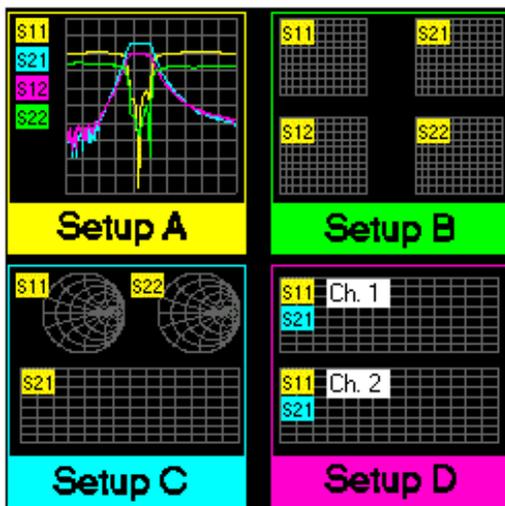
1. Press **DISPLAY**
2. then **[Measurement Setups]**

PNA Menu using a mouse

1. Click **Response**
2. then **Display**
3. then **Meas Setups**

No programming commands exist for this feature

The following are the four pre-configured measurement setups:



Arranging Existing Measurements

The following arrangements place EXISTING measurements into pre-configured Window arrangements using a sort algorithm.

How to select an Existing measurement arrangement

Using front-panel HARDKEY [softkey] buttons

1. Press **DISPLAY**
2. then [**Layout**]
3. then choose from the following...

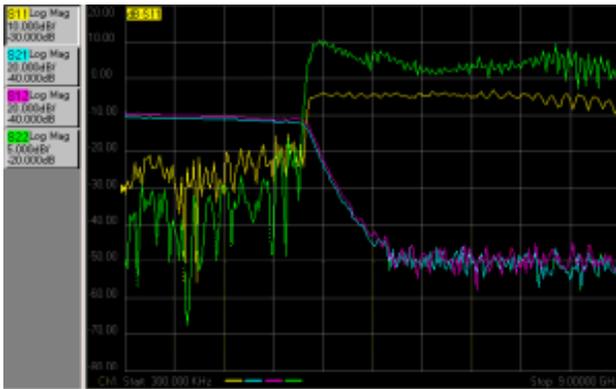
PNA Menu using a mouse

1. Click **Response**
2. then **Display**
3. then choose from the following...

◀ Programming Commands ▶

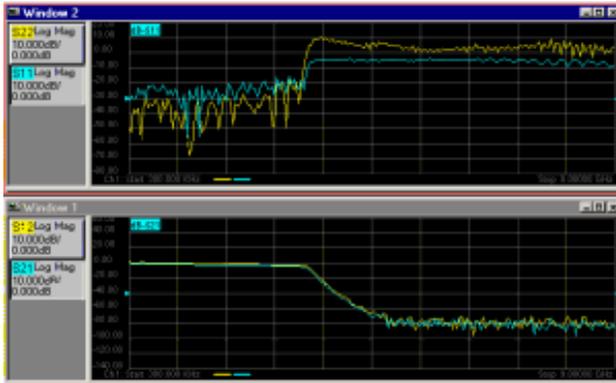
Overlay 1x

This configuration places all existing traces in a single window, all overlaid on each other.



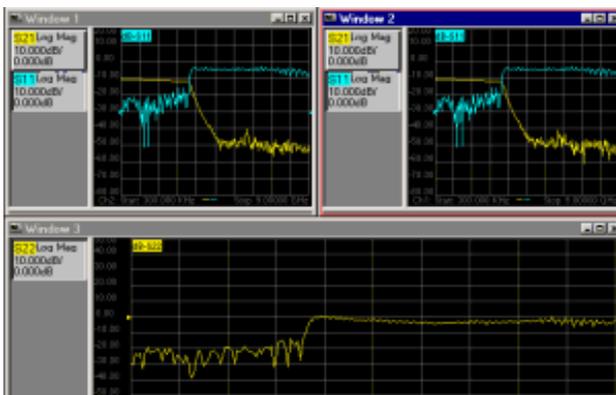
Stack 2x

This configuration places all existing traces in two "stacked" windows.



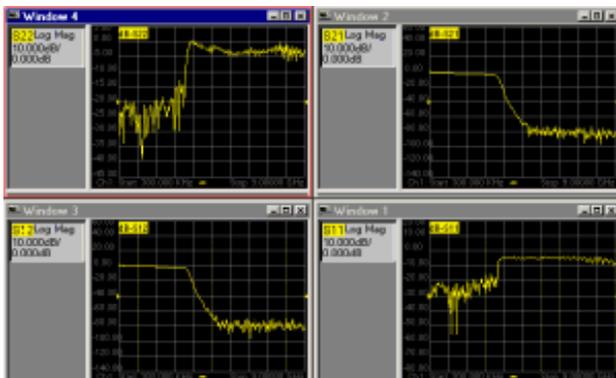
Split 3x

This configuration places all existing traces in three windows, two on top and one below.



Quad 4x

This configuration places all existing traces in four windows, one window in each screen quadrant.



Sort Algorithm

The sort algorithm for the Arrange Windows feature is designed to:

- Divide traces among windows based on their properties
- Group traces with common properties

The algorithm sorting is based on the following trace properties, in order of priority:

1. Format: circular (polar or Smith) versus rectilinear (log mag, lin mag, group delay, etc.)
2. Channel number
3. Transmission versus reflection

Note: The PNA [traces per window limitation](#) overrides this algorithm. An error occurs if the arrange selection cannot be completed with the current number of traces on the screen.

Last modified:

21-Jul-2011	New menu picks
3-Sep-2008	Removed legacy content
9/27/06	MX Added UI
9/12/06	Added link to programming commands

PNA-X RF Path Configurator

Allows you to configure hardware components that are available with selected [PNA-X options](#).

N522xA and N523x models do NOT have the RF Path Configurator.

How to access Path Configurator

Using HARDKEY [softkey] buttons:

1. Press TRACE/CHAN
2. then [Channel]
3. then [Hardware Setup]
4. then [Path Config...]

PNA Menu using a mouse:

1. Click **Trace/Chan**
2. then **Channel**
3. then **Hardware Setup**
4. then **Path Config...**

Programming Commands

The following image shows configuration with PNA-X Opt 423 (4-port, internal 2nd source, combiner, and mechanical switches). Your PNA-X may not include these options.

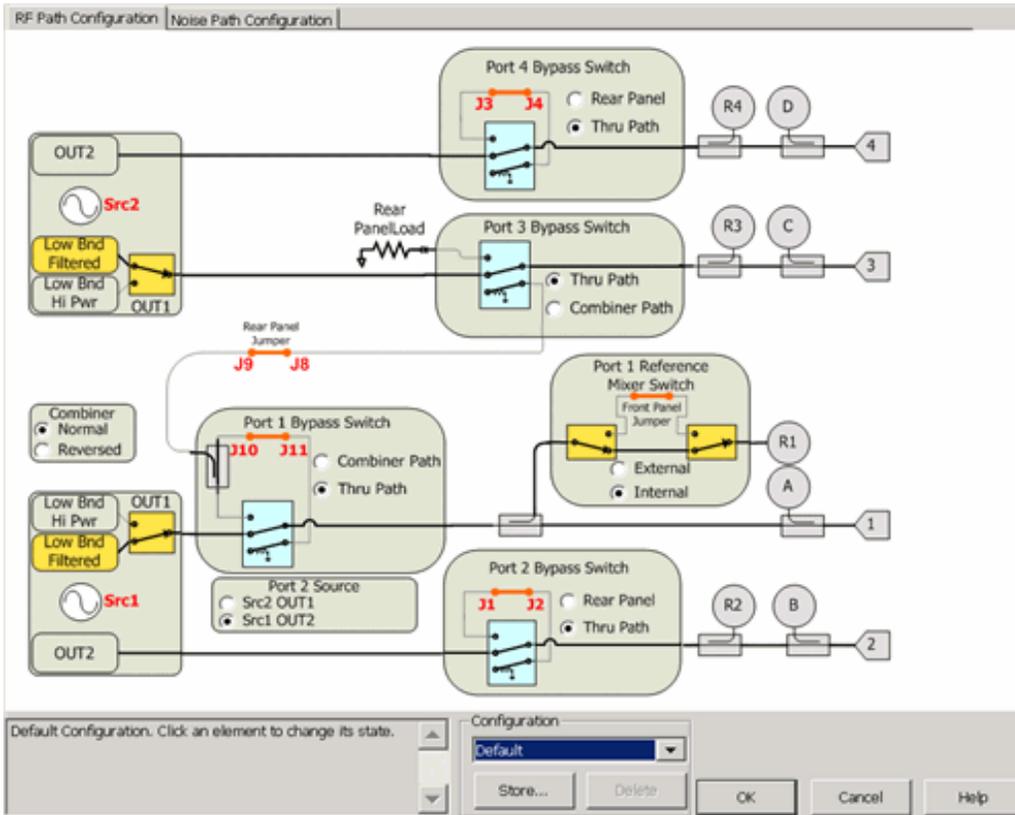
See Also

[IF Path Configuration](#)

[PNA-X specs](#) for block diagrams

[Internal Second Source limitations](#)

[PNA Configurations and Options](#)



Path Configuration dialog box help

See [Noise Figure tab](#) of the Path Configuration.

Note: With [selected PNA-X options](#), pulse modulation is available ONLY on **OUT1** of Src 1 and Src 2. [See block diagram.](#)

Different paths can be configured for each channel.

Configuration

Select, store, and delete factory configurations or user-defined configurations. Configurations are stored on the PNA hard drive.

Any configuration can be saved, and later recalled, from this dialog. Click **Store**, type a configuration name, then click **OK**.

Text area Displays text describing the physical connections required to complete the configuration. The text for factory configurations can NOT be edited. Text is saved as part of the configuration.

Cancel Closes the dialog and returns the configuration settings to the state they were in when the dialog was opened. Cancel does NOT undo **Store** and **Delete** actions that were performed while the dialog was open.

Notes

- Click or touch anywhere within a box on the screen to cycle through the available settings.
- Some switch settings alter graphics in areas other than where the switch is thrown.
- **If you don't hear switches clicking**, this could be why:
 - Electronic switches are **orange** on the path configuration dialog. These switches do not make noise when being thrown. Mechanical switches are **blue**.
 - The channel is not sweeping.
 - The following selections do NOT throw switches, but simply indicate how to connect jumper cables by drawing red lines on the dialog:
 - Combiner (Normal/Reversed)
 - Port 2 Source (Src2 OUT1/Src1 OUT2)

Note: To prevent premature wear, the PNA does not allow attenuators or other mechanical switches to switch continuously.

These mechanical devices are set for the entire channel. When more than one channel is used, and a mechanical device setting is NOT the same for all channels, only the ACTIVE channel is allowed to sweep. All other channels are NOT allowed to sweep (Blocked).

Learn how to [view the settings of all mechanical devices](#) in the PNA.

- Red lines are jumpers on the front or rear panel.
- **Src1 / Src2 Settings:**
 - **Low Bnd Filtered** (default setting) reduces harmonics below 3.2 GHz on OUT1 of both **Src1** and **Src2**.
 - **Low Bnd Hi Pwr** setting does not use the filtering which causes higher power below 3.2 GHz. See Max Leveled Power in the [specifications](#) for your PNA-X model.
 - Each source optionally has [pulse modulation](#) capability.
 - **Note:** Pulse modulation is available ONLY on **OUT1** of Src 1 and Src 2. [See block diagram](#). Your PNA must also have Pulse options. [Learn more](#).
- [Copy channel](#) feature also copies path configuration settings.
- Path Configuration is saved and recalled as part of an [instrument state](#).

Factory Configurations

Note: Recalling a stored configuration will over-write MANY RF and [IF path configuration](#) settings. Make your measurement settings AFTER recalling a stored configuration, NOT before.

Port 1 2-tone Routes Source 2 through the internal combiner to create a two-tone signal out port 1. The standard jumper configuration is used. This is the configuration that is used to make [IMD measurements](#).

Src 2 Out Port 2 (Option 423 ONLY) Routes Source 2 (OUT1) to port 2 using an external cable attached to the rear-panel J8 to J1. This allows source power at port 1 AND port 2 simultaneously. Source power is NOT available at port 3. [Learn more about Internal Second Source capability](#).

2 port Dual Source (Option 224 ONLY) Routes Source 2 (OUT1) to port 2 using an external cable attached to the rear-panel J8 to J1. This allows source power at port 1 AND port 2 simultaneously. This also allows pulsed measurements to be performed on both ports 1 and 2.

Hot S-parameters The rear panel jumpers must be manually reconfigured to allow the source 2 signal to be routed through the thru path of the internal combiner. Source 2 is used to drive the AUT into compression, so the highest possible output power is required. Source 1, routed through the coupled path of the combiner, is used for S-parameter measurements, so a small signal is sufficient. These two signals are combined but the frequencies are usually offset.

See Also

- [Configuration for High-power measurements](#)

Last modified:

5-Feb-2013	Added Src1/2 settings
18-Oct-2012	Added recall note
27-Apr-2011	Added Src 2 out Port 2 on 4-port (A.09.40)
23-Feb-2010	Added Blocked channel note
26-May-2009	Added factory configs
3-Sep-2008	Removed legacy content
14-Aug-2008	Added link to high-power measurements
5-Jan-2007	MX- New topic

Source Phase Control (Opt 088)

Option 088 allows you to control the phase of a PNA source or an external source. Two sources are required.

- [Overview](#)
- [Features and Limitations](#)
- [Phase Control Use-Cases](#)
- [How to make Phase Settings](#)
- [Calibration and Phase Control](#)
- [Active Load Pull Examples](#)

[See other 'Setup Measurements' topics](#)

Overview

The Source Phase Control feature provides a specific phase difference between two sources. The phase difference can be fixed (for example, at 90 degrees), or swept between two arbitrary phase values (for example, from 0 to 360 degrees).

This feature is allowed on ALL PNA-X models, and any N522xA model with either two internal sources or an external source and configurable test set.

Any combination of PNA internal or external sources may be used. One source is selected as the controlled source and the other source is the reference source. You select the two sources by selecting the ports at which the sources are available. The choice of ports is limited for you on the Phase Control dialog. [Learn about these limitations.](#)

In addition to selecting source ports, you also select the receivers to be used to measure the phase for the sources. This can be test port receivers or the reference receivers for the specified source ports. The receivers measure the relative phase of the sources, then adjust the phase of the controlled source, then remeasure until the phase difference is within the tolerance that you specify.

Phase can also be set without using any receivers. This is called 'Open Loop' mode. In this mode, the phase of the controlled source is set once, and iterations are not done, resulting in phase that is less accurate and stable compared to using receivers to measure and set phase. Use Open Loop mode when you need to use the receivers to measure other parameters.

The phase of the controlled source can be swept relative to the reference source. The phase difference

between the controlled and reference source is incremented and iterated on consecutive data points. Before starting the Phase Control dialog, select Sweep Type = Phase Sweep. [Learn how](#).

When the phase of a source is controlled, the power of that source is also controlled using [Receiver Leveling](#). Instead of the normal receiver-leveling mode where only one receiver is used, when phase control is active the ratio of two receivers is used to level the power of the controlled source. This is useful for making active load-pull measurements as described below. In Open Loop mode, neither the phase nor the power of the source is controlled.

Features and Limitations

- Phase Control is allowed ONLY in a standard [S-parameter](#) channel.
- Phase Control can be used with [Wideband pulse measurements](#) - NOT in narrow-band pulse mode.
- [Point Averaging](#) is NOT allowed.
- **External sources** are supported. Learn how to [Configure an External Device](#). Phase can be controlled on Agilent MXG, PSG, ESG and EXG sources. The external source must be routed through the rear panel so that a reference receiver can measure its phase. Use the [Path Configuration dialog](#) to make switch settings and [enable FOM mode](#).
- [Remote commands](#) are available that allow the phase and power of each point to be set individually, much like in source power calibration. Use these commands if you need to create a specific pattern of amplitude/phase states, such as characterizing the load-pull of an amplifier.

Phase Control Use-Cases

Phase control and phase sweep is useful in the following applications:

Active load control

Provide a controlled, electronically-settable impedance to the output port of a DUT under fixed or swept-frequency conditions. Some examples are: measuring the gain and output power of an amplifier with a known load, and measuring the output from a directional detector with a known load.

The reference source is applied to the DUT input port, and the controlled source is applied to the DUT output port as a reverse input wave. The phase and power level of the controlled source is set relative to the forward output wave of the DUT (which is determined by the reference source), so that any arbitrary load impedance (γ) can be set.

Optionally, the phase of the controlled source can be swept with a constant frequency, so that the phase of γ rotates while the magnitude of γ remains constant. The ratio of reverse input wave and forward output wave as viewed on a Smith chart or polar display would appear as a circle. This capability can be combined with external load-pull software to create traditional load-pull power contours.

Phase-controlled sources

Set the phase and magnitude of one source relative to a reference source, to provide differential, quadrature, or arbitrary phase-offset signals at a fixed or swept frequency. Typically, another instrument, receiver, or detector would be required to measure the response of the DUT.

How to make Phase Control settings:

Using front-panel HARDKEY [softkey] buttons

1. Press **SWEEP**
2. then **[More]**
3. then **[Phase Control]**

Using a mouse with PNA Menus

1. Click **Stimulus**
2. then **Sweep**
3. then **More**
4. then **Phase Control**

To perform a Phase Sweep:

1. Press **SWEEP**
2. then **[Sweep Type]**
3. then **[Phase]**

1. Click **Stimulus**
2. then **Sweep**
3. then **Sweep Type**
4. then **Phase Sweep**

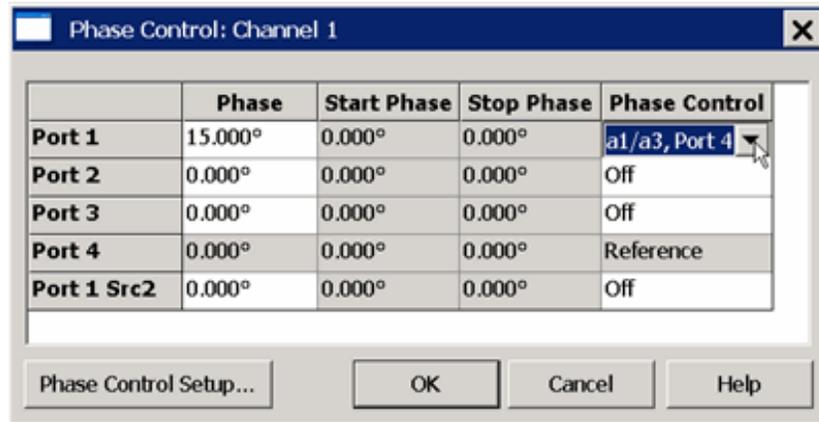
To select (view) a Phase format: ([Learn more](#))

1. Press **FORMAT**
2. then **[Phase]**

1. Click **Response**
2. then **Format**
3. then **Phase, Unwrapped Phase, or Smith**

Programming Commands

Phase Control dialog box help



Phase Specify any Fixed Phase setting.

Start / Stop Phase Available when Sweep Type is set to Phase Sweep. [Learn how](#). Enter the Start and Stop phase values in degrees.

Phase Control Click in the port cell to be controlled, then choose from the following:

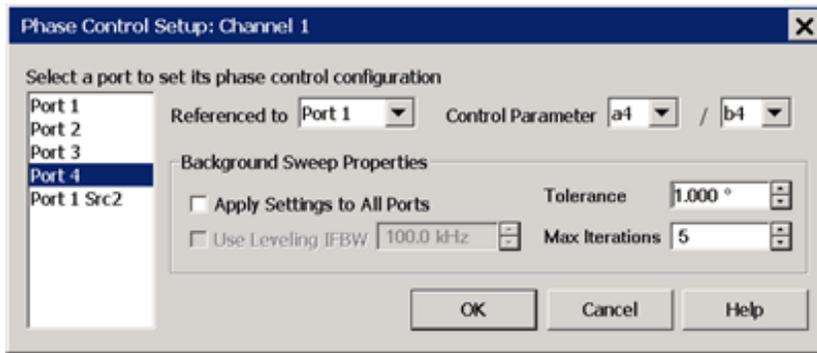
- **Off** - Phase is NOT set or controlled.
- **Open Loop** - Phase and power is set, but receivers are NOT used to measure and iterate the phase or power of the source. Therefore, the setting of phase is not as accurate or stable. Open Loop mode can be used with phase sweep (for example, from 0 to 360 degrees). However, each sweep may not start at 0 degrees. NO settings on the following Phase Control Setup dialog are used in Open Loop.

Note: After selecting Open Loop, set each source to **ON** (not Auto) using the [Power and Attenuators dialog](#).

- **<rec/rec>, reference port (Controlled)** - Phase and power is measured and iterated to within the specified tolerance. Click **Phase Control Setup** to specify the source ports, the receivers, and iteration properties to use to control phase. Use the [Receiver Leveling](#) feature to set the power iteration properties. The source or receiver being leveled can NOT be changed in the Receiver Leveling dialog.
- **Reference** is displayed to indicate the specified reference port. You can also perform receiver leveling on the reference source. [Learn how](#).

Phase Control Setup Click to launch the Phase Control Setup dialog.

Phase Control Setup dialog box help



Select a port to set its phase control configuration This is the 'controlled' port.

Referenced to Select a source port to be used as a phase reference for the controlled port.

The two internal PNA sources are available ONLY at specific ports. These choices are limited for you on the Phase Control dialog. For example on a 4-port PNA-X, the possible port pairings are: 1/3, 1/4, 2/3, or 2/4. Port 1 can NOT be paired with Port 2, and Port 3 can NOT be paired with Port 4. [Learn more about these limitations.](#)

Control Parameter Select the receivers to be used to measure the phase and power of the sources.

- The LEFT receiver (a4 in the above image) measures the controlled source.
- The RIGHT receiver (b4 in the above image) measures the reference source.

The swept phase or phase offset will be the difference between these two receivers. Use either standard notation (A, R1) or logical receiver notation (a1,b1). [Learn more.](#)

Select the receivers based on your application. You are responsible to make sure that your DUT configuration routes the signals of interest to the correct receivers. Otherwise, the phase will not be properly controlled. For example, if you select the configuration in the above dialog, both Port 4 (controlled source) and Port 1 (reference source) must be connected and measured by the Port 4 (a4 and b4) receivers. This would typically be at your DUT output. See the [Active Load Pull Example](#) below.

Background Sweep Properties

Background sweeps are phase and power measurements that are made, but the results are not displayed. For each data point, when subsequent measurements are within the specified tolerance, that point is considered settled. If consecutive phase or power measurements of the same data point are NOT within the specified tolerance before the Max Iterations is reached, then one of the following messages are displayed:

- Phase leveling warning: phase not settled.

- Phase leveling warning: power not settled.
- Phase leveling warning: phase and power not settled.

Apply Settings to All Ports When checked, the specified settings are used for all background sweeps for all phase-controlled ports. When cleared, the following three settings are specified independently for each port pair.

Use Leveling IFBW (NOT available for this release).

Tolerance When consecutive phase measurements of the same data point are within this value of each other, then the phase measurement is considered settled.

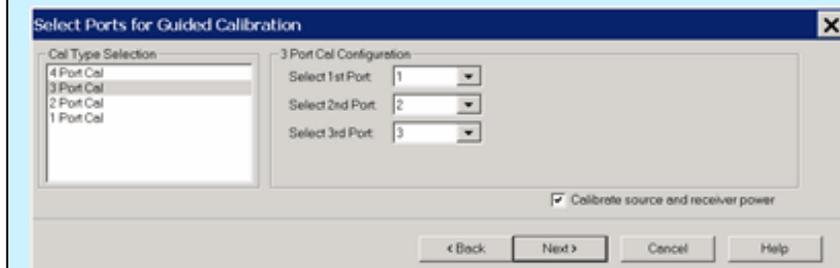
Max Iterations Sets the maximum number of background phase measurements to perform in order to achieve settling. If the phase is not sufficiently settled after these measurements, then the closest value is used.

Calibration and Phase Control

Calibrate only those ports that are used for a phase control measurement. For example, if using ports 1 and 3 for phase control, then do NOT calibrate all four ports. If other ports are calibrated, then even ports 1 and 3 may not yield acceptable results.

Important

During the Guided Calibration, at the Select Ports dialog, check the [Calibrate source and receiver power](#) checkbox. Otherwise, the calibration may not be accurate.



After performing calibration, the phase is aligned and the power is accurate at the calibration plane.

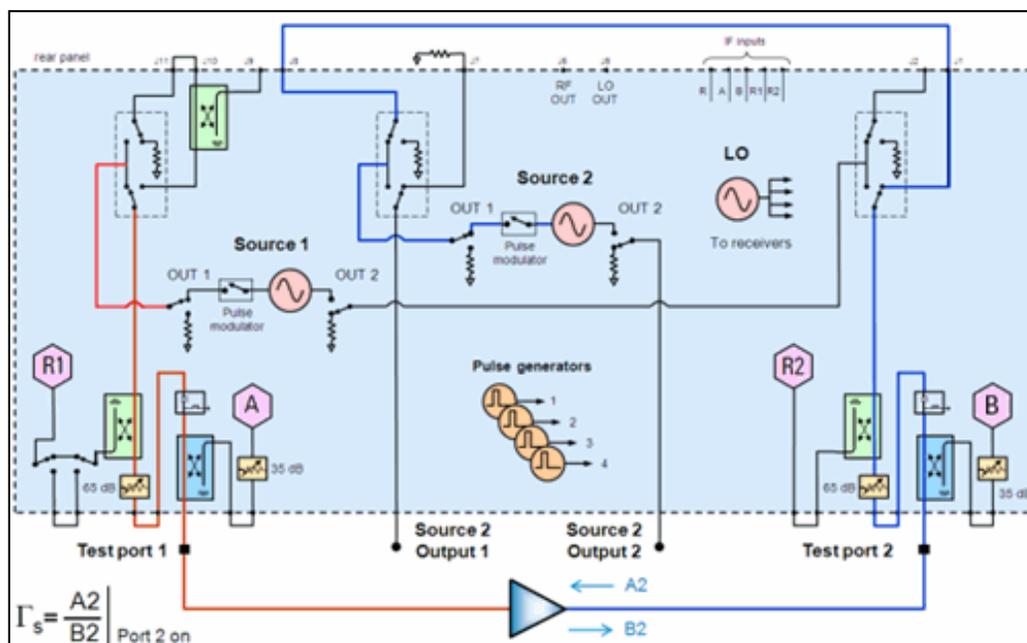
Active Load Control Example (4-port PNA-X)

[See setup using a 2-port 2-source PNA-X.](#)

<input checked="" type="checkbox"/> Power ON (All Channels) <input type="checkbox"/> Port Powers Coupled		Name	State	Port Power	Start Power	Stop Power	Auto Range	Source Atten.	Leveling Mode
Port 1	Auto	-5.00 dBm	-10.00 dBm	0.00 dBm	<input checked="" type="checkbox"/>	0 dB	Internal		
Port 2	Auto	-5.00 dBm	-10.00 dBm	0.00 dBm	<input checked="" type="checkbox"/>	0 dB	Internal		
Port 3	Auto	-5.00 dBm	-10.00 dBm	0.00 dBm	<input checked="" type="checkbox"/>	0 dB	Internal		
Port 4	Auto	-2.00 dBc	0.00 dBc	0.00 dBc	<input type="checkbox"/>	10 dB	a1/b1		

Active Load Control with 2-Port 2-Source PNA-X Models

1. Connect a rear-panel jumpers cable from J8 to J1.
2. On the Phase Control Setup dialog, select **Port 2** as the controlled source and **Port1 Src 2** as the reference source.
3. On the [Path Configuration dialog](#), on the Configuration box, select **2- port dual source**.
4. Select the Control Parameter (receivers) **a2 / b2**.



Last Modified:

- 16-Oct-2013 Added Guided Power Cal required
- 29-Aug-2012 Added EXG
- 9-Apr-2012 Edited allowed models
- 20-Dec-2010 MX New topic

Customize the PNA Screen

You can customize your PNA screen by showing or hiding the following display elements. All of these selections are made from the PNA **Response > Display** menu.

- [Layout](#) (Separate topic)
- [Windows](#) (Separate topic)
- [Measurement Setups](#) (Separate topic)
- [Display Labels](#)
- [Marker Display](#) (Separate topic)
- [Tables](#)
- [Toolbars](#)
- [Status and Title Bars](#)
- [Display Colors](#) (Separate topic)
- [Grid: SOLID | Dotted](#)
- [System Date and Time](#)
- [Minimize Application](#)

See Also

[Expanded display capabilities of the PNA-X](#)

[Traces, Channels, and Windows on the PNA](#)

How to show/hide the Status Bar and Title Bars

Using front-panel

HARDKEY [softkey] buttons

1. Press **DISPLAY**
2. then [**Tables and Toolbars**]
3. then [**Status Bar or Title Bars**]

PNA Menu using a mouse

1. Click **Response**
2. then **Display**
3. then **Toolbars**
4. then **Status Bar or Title Bars**

[Programming Commands](#)

Status Toolbar

Status CH 1: S11 No Cor

When enabled, the status bar is displayed along the bottom of the PNA screen. The primary status bar shows the following:

TIP: Right-click on many of these items in the status bar for quick access to settings.

- [Channel Trigger State](#) (Cont, Groups, Single, Hold)
- Active channel
- Measurement parameter for the active trace
- [Trace Math](#)
- [Error correction](#) for the active trace
- [Averaging Factor](#) for the active channel
- [Smoothing](#) Percentage
- [Transform](#) (On)
- [Gating](#) (On)
- IF Gating Enabled for [Pulsed App](#): (G)
- Manual IF Filtering for Pulsed App: (F)

- Delay if invoked using [Phase Offset](#), [Electrical Delay](#), or [Port Extensions](#).
- Loss if invoked using [Magnitude Offset](#) or [Port Extensions](#).
- GPIB status: Local (LCL), Remote Talker Listener (RMT), or System Controller (CTL).
- Error Status: (LVL, LCK, etc)
- System Date and Time - Can be set ON or OFF.

Note: A second level status bar appears when using [External Test Set Control](#) or [Interface control](#).

The status bar state (ON or OFF) will not change when the PNA is Preset.

Title Bars



The Title bar shows the window number and Minimize / Maximize icons.

- Checked - Title bars for all PNA windows are shown.
- Cleared - Title bars for all PNA windows are hidden. This allows more room to display measurement results.

Toolbars

You can display different toolbars to allow you to easily set up and modify measurements.

How to display Toolbars

Using front-panel

HARDKEY [softkey] buttons

1. Press **DISPLAY**
2. then **[Tables and Toolbars]**
3. then **[Toolbars]**

PNA Menu using a mouse

1. Click **Response**
2. then **Display**
3. then **Toolbars**

Programming Commands

List of toolbars

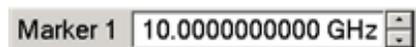
- [Active Entry](#)
- [Softkeys](#)

- [Markers](#)
- [Time Domain](#)
- [Port Extension](#)
- [All Off](#) (NOT on softkeys)

Note: There is also a Cal Set toolbar available for [Monitoring Error Terms](#)

Beginning with PNA Rev. A.10.0, the Measurement, Sweep Control, and Stimulus toolbars were eliminated.

Active Entry Toolbar



When used with softkeys, this area allows numeric values to be entered for settings. From the keyboard, enter G for Giga, M for Mega or milli, K for kilo, and so forth.

Softkeys

Softkeys are automatically turned ON when one of the 'function' hardkeys is pressed. This setting allows you to turn the softkeys OFF to show more measurement space on the screen. The softkeys will reappear when another function hardkey is pressed.

Markers Toolbar



The markers toolbar allows you to set up and modify markers. It shows:

- Marker number
- Stimulation value
- Marker functions:
 - Delta
 - Start/Stop
 - Center/Span

Tip: To use the Front Panel Knob to change marker position, first click the **Stimulus** field of the marker toolbar. Then turn the knob.

[Learn more about Markers](#)

Time Domain toolbar

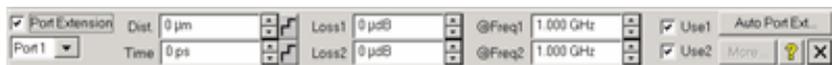


The Time Domain toolbar allows you to do the following:

- Turn Transform and Gating ON / OFF
- Change the Start / Stop times for both Transform and Gating
- **More...** launches the [Time Domain Transform](#) dialog box
- **X** Closes the toolbar

The front panel **Tab** key steps through all of the settings on all of the toolbars on the display. If Tab does not work, press one of the Active Toolbar (color) keys.

Port Extension toolbar



The Port Extension toolbar allows you to set Port Extensions while viewing the measurement trace. Learn more about [Port Extensions](#).

All Off

This allows you to **hide all toolbars** with a single selection. NOT available on sofkeys.

Tables

Tables are displayed at the bottom of the selected window.

How to display tables

Each window can display only one table at a time.

Click **Hide Table** to turn OFF the table.

Using front-panel

HARDKEY [softkey] buttons

1. Press **DISPLAY**
2. then **[Tables and Toolbars]**
3. then **[Tables]**

PNA Menu using a mouse

1. Click **Response**
2. then **Display**
3. then **Tables**

 **Programming Commands** 

List of tables

- [Marker Table](#)
 - [Limit Line Table](#)
 - [Segment Table](#)
-

Marker Table

You can display a table of marker settings. These settings include the:

- Marker number
- Marker reference (for delta measurements)
- Frequency
- Time and Distance (for Time Domain measurements)
- Response

Learn more about [Markers](#)

Limit Line Table

You can display, set up, and modify a table of limit test settings. These include:

- Type (MIN, MAX, or OFF)

- Beginning and ending stimulus values
- Beginning and ending response values

Learn more about [Limit Lines](#).

Segment Sweep Table

You can display, set up, and modify a table of segment sweep settings. These include:

- State (On/Off)
- Start and Stop frequencies
- Number of Points
- IF Bandwidth (if independent levels)
- Power Level (if independent levels)
- Sweep Time (if independent levels)

Learn more about [Segment sweep](#)

Display Labels

How to show and hide Display items

Using front-panel HARDKEY [softkey] buttons

1. Press **DISPLAY**
2. then [**Labels**]

PNA Menu using a mouse

1. Click **Response**
2. then **Display**
3. then **Labels**

[Programming Commands](#)

List of Display Labels

- [Window Title](#)
- [Trace Title](#)

- [Trace Status](#)
 - [Frequency, Stimulus](#)
-

Window Title

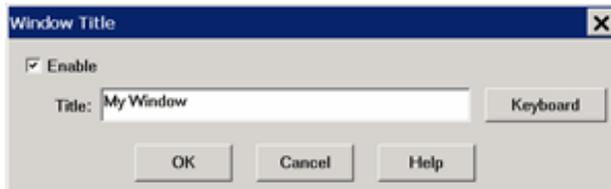
You can create and display a title for each **window**.

- The limit is set by the number of windows that are displayed.
- The title (My Window) is annotated in the upper-left of the window as follows:



How to enter a Window Title

1. Click in the window to make the window active.
2. Click **Response**, then **Display**, then **Labels**, then **Window Title**.



3. Click **Enable**, then type the trace title. Click **Keyboard** to type with a mouse.
 4. To remove the trace title, clear the **Enable** checkbox, or delete the text from the dialog entry.
-

Trace Title

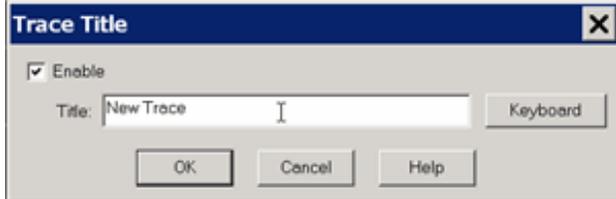
A Trace Title overwrites the Measurement Parameter in the [Trace Status](#) area, the [Status Bar](#), and [hardcopy prints](#).

- This title has priority over [Equation Editor](#) titles.
- The practical limit is about 70 characters if there is only one trace.
- Spaces are accepted but not displayed; use underscores.
- The title is annotated as follows:

A screenshot of a trace title bar. The text "Tr 1 New Trace LogM 10.00dB/ 0.00dB" is displayed in yellow on a black background.

How to enter a Trace Title

1. Click the [Trace Status](#) label to select a trace.
2. Click **Response**, then **Display**, then **Labels**, then **Trace Title**.



3. Click **Enable**, then type the trace title. Click **Keyboard** to type with a mouse.
4. To remove the trace title, clear the **Enable** checkbox, or delete the text from the dialog entry.

Trace Status

	<p>Trace status is annotated at the top of each window.</p> <p>The highlighted trace number indicates the Active Trace.</p> <p>Click the title to select a trace.</p>
---	---

Trace Status shows the following:

- Trace number (Tr x). This is the trace number of the channel; NOT the window trace number which is used in many programming commands.
- Measurement parameter. This can be replaced with a custom **Trace Title** (see below).
- Format
- Scaling factor
- Reference level

[How to show/hide Trace Status](#)

Frequency/Stimulus



Frequency/stimulus information is displayed at the bottom of each window on the screen. It shows:

- Channel number
- Start value
- Stop value

[How to show/hide Frequency/Stimulus information](#)

How to set PNA Grid and Clock display settings

Using front-panel

HARDKEY [softkey] buttons

1. Press **DISPLAY**
2. then [**More**]

PNA Menu using a mouse

1. Click **Response**
2. then **Display**
3. then **More**

[Programming Commands](#)

Grid: **SOLID** | Dotted

Set whether to display ALL open window grid lines in solid or dotted lines. The selected setting is shown in CAPS. Once set, new windows are created using this setting. Grid lines return to **SOLID** when the PNA is Preset.

Set the color of the grid using [Display Colors](#).

[How to display grid settings](#)

System Date and Time (also known as Clock)

The PNA system date and time can be shown in the far right corner of the status bar.

The format is: year-mo-day hr:min and can NOT be changed.

[Learn how to set the PNA time](#)

[How to show/hide the PNA clock](#)

Minimize Application

The Network Analyzer application can be minimized to show the desktop and Windows taskbar.

1. Click **File**
2. then **Minimize Application**

To restore the PNA application, click the PNA application on the Windows taskbar.

Last modified:

8-Nov-2013	Added clock
21-Jun-2013	Removed 3 toolbars (10.0)
3-May-2012	Removed C models
21-Jul-2011	Several changes (9.5)
16-Mar-2010	Added Grid lines (9.2)
3-Sep-2008	Removed legacy content
27-Aug-2007	Edited readout section
9/12/06	Added link to programming commands
9/27/06	MX Added UI

Copy Channels

Copy channels allows you to make a duplicate channel of the same [Measurement Class](#) and with the same stimulus conditions as an existing channel.

- [Why Copy Channels](#)
- [How to Copy Channels](#)
- [List of Channel Settings](#)

Note: Beginning with A.09.40, Copy Channels CAN be used with [PNA Applications](#), such as FCA, Gain Compression, or Noise Figure.

Other Setup Measurements Topics

Why Copy Channels

Copy channel settings if you need to create several channels that have slightly different settings.

For example, if you have an amplifier that you want to characterize over a frequency span with several different input power levels.

Follow these steps:

1. Create one measurement with your optimized channel settings.
2. Copy that channel to new channels.
3. Change the power level on the new channels.

The alternative to using Copy Channels is to create new default measurements on new channels. Then change every channel setting to your new requirement. This is very time consuming and thus shows the benefit of the Copy Channels feature.

How to Copy Channels

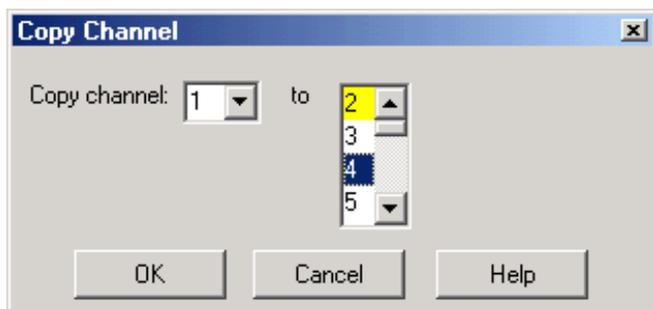
Using front-panel HARDKEY [softkey] buttons

1. Press TRACE/CHAN
2. then [Channel]
3. then [More]
4. then [Copy Channel]

PNA Menu using a mouse

1. Click **Trace/Chan**
2. then **Channel**
3. then **Copy Channel**

Programming Commands



Copy Channel dialog box help

Copies an existing channel's settings to another channel. Measurement traces from the source channel are NOT copied.

Copy channel (also known as '**Source**' channel): Select a channel to copy.

to (also known as '**Destination**' channel): Scroll to select a channel to copy settings to. Compatible channel numbers that are currently being used are highlighted. They can be selected and overwritten.

The following are compatible destination channels:

- A channel that does not yet exist. The new channel is created with the channel's default measurement.
- A channel of the **same** [Measurement Class](#) as the source. The existing measurements remain on the destination channel.
- A channel of any Measurement Class that contains no measurements. Again, the destination channel is created with the channel's default measurement.

Notes:

- You can copy channel settings to ONLY one new or existing channel. Repeat this operation to copy to more than one channel.
- The source channel is ALWAYS copied to the Active window. If you want the destination channel in a separate window, first create a compatible new measurement in a new window. Then make sure it is the Active window before you copy the channel into it.
- The measurement in the destination channel becomes the active measurement.

For example:

1. **Source** channel 1: Standard S21 measurement
2. **Destination** NEW channel 2
3. **Result:** Source channel 1, S21 Measurement AND channel 2, S11 measurement. Both with same stimulus settings and in the same window. Channel 2, S11 measurement is the active measurement.

For more information see [Traces, Channels, and Windows on the PNA](#)

List of Channel Settings

- [Frequency Span](#)
- [Power](#)
- [Cal Set usage](#)
- [Source Power Cal data](#)
- [IF Bandwidth](#)
- [Number of Points](#)
- [Sweep Settings](#)
- [Average](#)
- [Trigger \(some settings\)](#)

Last modified:

12-Mar-2014	Works with Apps (A.09.40)
13-Feb-2008	Added note about Apps
9/12/06	Added link to programming commands

DC Source Control

When a DC Source (power supply) is configured as an External Device, the new DC source can be controlled from the PNA using this dialog. [Learn how to configure the DC Source as an External Device.](#)

- [How to start the DC Source Control dialog](#)
- [The DC Source Control dialog box](#)
- [The DC Limits dialog box](#)

Other Setup Measurements Topics

How to start the DC Source Control dialog

To set the X-axis to display the DC source, click **Response, Display, Labels, Select X-Axis**, then select the DC Source to display.

Using front-panel

HARDKEY [softkey] buttons

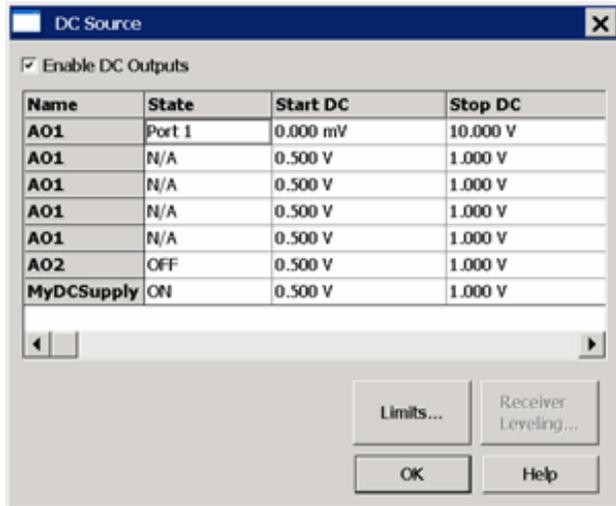
1. Press **SWEEP**
2. then **[More]**
3. then **[DC Source]**

PNA Menu using a mouse

1. Click **Stimulus**
2. then **Sweep**
3. then **DC Source**

◀ Programming Commands ▶

DC Source Control dialog box help



Name Lists the names of the configured DC Sources. In the above image:

- **A01** and **A02** are internal PNA DC sources that are available on the rear-panel Power I/O connector (Pins 3 and 4). [Learn more.](#)
- **MyDCSupply** is the name of an external DC Source. [Learn how to setup and configure an External DC Source and DC Meter.](#)

State Set the state of the DC source.

- **ON** DC Source is always ON.
- **OFF** DC source is always OFF.
- **Per Port** The Name selection for that DC source expands to allow an Port <n> / N/A setting for each PNA port. When the RF source for that port <n> is ON, then the DC source for port <n> is also ON. Select 'N/A' to turn the DC Source OFF for that port.

For example: In the above image...

- A01 will be ON when the RF source at Port 1 is ON, stepping from 0 to 10V.
- A01 will be OFF when all other RF source ports are ON.
- A02 will always be OFF
- MyDCSupply will always be ON, stepping from .5V to 1.0 V each sweep.

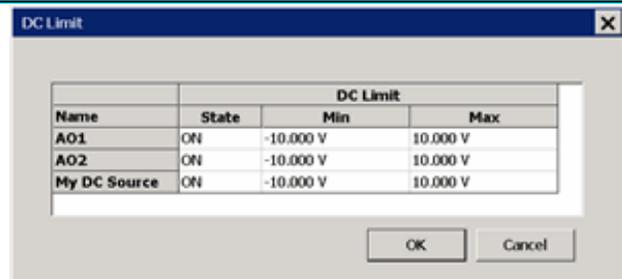
Start / Stop DC Set the start and stop voltages of the DC source. The PNA will step the voltage of the DC source from Start to Stop in increments = $(\text{Stop} - \text{Start}) / \text{Number of data points}$.

Buttons

Limits Click to start the DC Limits dialog.

Receiver Leveling For future use.

DC Limits dialog box help



Select the Minimum and Maximum voltages to which the specified DC sources can be set by the PNA.

Last modified:

10-Jan-2012 New topic

ADC Measurements

The PNA is equipped with two ADC (Analog to Digital Converter) inputs on the rear-panel [Power I/O connector](#) (Pins 7 and 8). These ADC inputs can be used as measurement receivers and display measurements on the PNA screen.

- Analog Inputs (AI1 and AI2) can be used for measuring from -10V to +10V. These inputs can be considered auxiliary receivers and used in a similar way as S-Parameter receivers.
- Analog Output Sense inputs (AOS1 and AOS2) can be used to measure the corresponding DAC outputs.
- Analog Ground input (AIG) can be used to measure the instruments analog ground (PNA-X only).

Two DAC outputs are supplied on the [Power I/O connector](#) (Pins 3 and 4). These DAC outputs are controlled from the [DC Source Control dialog](#).

[Learn how to configure an external DC Source or DC Meter.](#)

How to create ADC receiver measurements

1. Press **TRACES**
2. then **[New Traces]**
3. then **[Receivers tab]**

1. Click **Trace/Chan**
2. then **New Trace**

Programming Commands

New Trace (ADC) dialog box help



Note: Sweep speed slows dramatically when measuring more than two ADC receivers.

On the [New Trace dialog](#), click the **Receivers** tab.

Activate - check any empty line to create a trace.

Numerator - select from the following:

- **AI1** or **AI2**- Input 1 or 2
- **AOS1** or **AOS2** - Output sense 1 or 2
- **AIG** - Analog ground (PNA-X only)

Denominator - NOT available (ONLY unratiod measurements)

Source Port - The ADC receiver is measured when the specified source port is sweeping. Select None to always measure the ADC receiver.

ADC receiver traces are labeled as shown in the following images:



- The ADC1 input is being measured, with 2 as the source port.
- The Y axis is U (unitless).
- The default trace [format](#) is Real (linear).
- To change the X-axis to display the ADC parameters, click **Response**, then **Display**, then **Labels**, then **Select X-Axis**, then select the ADC.

ADC Traces and other useful PNA functions

Although most PNA functions work with ADC traces, the following may be especially useful.

- [Equation Editor](#) can be used with the trace data. Although the PNA-X ADC is measuring voltage (-10V to +10V range in 14 bits), by using a trace formula, this voltage can represent other types of measurement parameters (such as current, temperature, or a scaled voltage). [See PAE example.](#)
- [Trace averaging](#) and [Trace Smoothing](#) can be used to remove trace noise.
- [Dwell time](#) can be used to allow for settling.

PNA Functions Not Supported

- Calibration for ADC receivers is NOT supported.
 - Not supported in [Noise Figure application](#)
-

Last Modified:

2-Aug-2012	Added noise figure
21-Feb-2012	Added DAC outputs and DC source control
31-Aug-2009	Fixed two typos
23-Mar-2009	Added pin numbers
19-Apr-2007	MX New topic

Undo/Redo Settings

If you make an incorrect setting, you can quickly recover by selecting Undo. If you then incorrectly Undo a setting, you can Redo the undone setting.

- Undo and Redo applies ONLY to [selected PNA settings](#).
- The Undo stack remembers 16 levels of Undo-able settings.

How to Undo or Redo a setting

Tips:

- Click or touch the Undo and Redo Icons:



Undo Redo

- Undo/Redo can be stored to a User Key or Favorite softkey. [Learn more](#).
- With a mouse, right-click on the Softkeys or on the Entry toolbar.
- With a keyboard:
 - Undo....Ctrl+Z
 - Redo....Ctrl+Y

Using front-panel HARDKEY [softkey] buttons

1. Press **SYSTEM**
2. then [**Undo**]
3. then [**Undo**] <setting>
or [**Redo**] <setting>

PNA Menu using a mouse

1. Click **Utility**
2. then **Undo**
3. then **Undo**<setting>
or **Redo** <setting>

SCPI and COM programming and Undo/Redo:

- There are NO Programming commands to invoke Undo/Redo
- Programing commands are NOT Undo-able.

- The Undo stack is cleared when programming commands are sent to the PNA.

Clear Undo History

To clear the Undo stack, press **System**, then **Service**, then **Utilities**, then **Click Undo History**.

Undo and Security

- Undo/Redo is disabled with **High** and **Extra** security levels. [Learn more](#).
- State files that are saved for Undo/Redo purposes (for example: Preset) are deleted when any of the following occur:
 - The PNA Security level is changed
 - The PNA App is started or closed.

Selected Undo-able settings

You can Undo or Redo the following **PNA settings**:

Note: There are several PNA settings that are NOT Undo-able. Because of this, when you attempt to Undo a long sequence of operations, it is unlikely that the original state can be recreated exactly.

- [Preset](#)
- [File Recall](#)
- Frequency Settings:
 - For Standard Class, Gain Compression, and NF: Start, Stop, Center, Span, CW
 - For SMC, VMC: Mixer Setup dialog Apply
 - For Swept IMD: Freq softkeys, setup dialog Apply
 - IM Spectrum: Freq softkeys, setup dialog Apply
 - Noise Figure: Freq softkeys, setup dialog Apply
 - GCX, IMDX, NFX: setup dialog Apply
- [Turn off Marker](#) and [Marker All OFF](#)
- [Number of Points](#)
- [Power Level](#) - most applications and S-parameters
- [Add or Change Measurement Class](#)

- [Turn OFF Channel](#)
 - [Close Window](#)
 - [New Channel , new Window, and new Trace.](#)
 - Delete Trace
 - Window Tile
 - Change Layout (1x, 2x, 3x, 4x)
 - Move Trace, Drag Trace
 - Zoom XY, Zoom Out Full
 - Autoscale All, Autoscale
 - Scale, Reference Level, Reference Position
 - Scale Coupling dialog
 - Electrical Delay
 - Phase Offset
 - Measurement Setups dialog
 - Format
 - Sweep Type
 - Data->Memory
 - Single Marker Searches (Max, Min, Target, Peak...)
 - Multi-marker Searches (Bandwidth, Power Saturation, Normal Operating Pt)
 - Change a Marker's stimulus value: softkeys, dialog or drag
 - Change cell in Segment Table
 - Mechanical Settings dialog
-

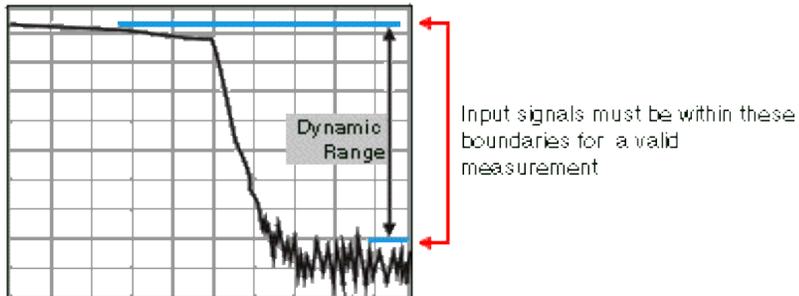
Last modified:

11-Mar-2013	Removed Re-save state
6-Feb-2012	New topic

Dynamic Range

Dynamic range is the difference between the analyzer receiver's maximum input power and the minimum measurable power (noise floor). For a measurement to be valid, input signals must be within these boundaries.

Increasing dynamic range is important if you need to measure very large variations in signal amplitude, such as filter bandpass and rejection. The dynamic range is shown below for an example measurement.



To help reduce measurement uncertainty, the analyzer dynamic range should be greater than the response that the DUT exhibits. For example, measurement accuracy is increased when the DUT response is at least 10 dB above the noise floor. The following methods can help you increase the dynamic range.

- [Increase the Device Input Power](#)
- [Reduce the Receiver Noise Floor](#)
- [Improving Dynamic Range using the Front-Panel Jumpers](#)

[Other topics about Optimizing Measurements](#)

Increase Device Input Power

Increase the DUT input power so that the analyzer can more accurately detect and measure the DUT output power. However, use caution - too much power can damage the analyzer receiver or cause compression distortion.

Caution! Receiver input damage level: +15 dBm.

[See how to increase input power to the device](#)

Tip: You can further increase dynamic range by using an external booster amplifier to increase the input power to the DUT. See [High Power Amplifier Measurements](#).

Reduce the Receiver Noise Floor

You can use the following techniques to lower the noise floor and increase the analyzer's dynamic range.

- Reduce crosstalk between the PNA receivers when measuring signals close to the noise floor. See [Receiver Crosstalk.](#))
- Use **Sweep Averaging** - learn more about [Sweep Average](#)
- Reduce the **IF Bandwidth** - learn more about [IF Bandwidth.](#)
- In [Segment sweep](#) mode each segment can have its own IF bandwidth. For example, when measuring a filter:
 - In the passband, the IF bandwidth can be set wider for a fast sweep rate, as long as high-level trace noise is kept sufficiently small.
 - In the reject band, where noise floor contributes significantly to measurement error, the IF bandwidth can be set low enough to achieve the desired reduction in average noise level.

Improving Dynamic Range using the Front-Panel Jumpers

Direct Access

The simplest method to improve dynamic range is to remove a RCVR 'n' IN [front-panel jumper](#) and route the DUT output directly into that PNA receiver. This bypasses the directional coupler and limits the ability to provide Full Error Correction because the signal can not be applied in the reverse direction.

Refer to the [PNA specifications](#) to learn the dynamic range that is available with direct receiver access.

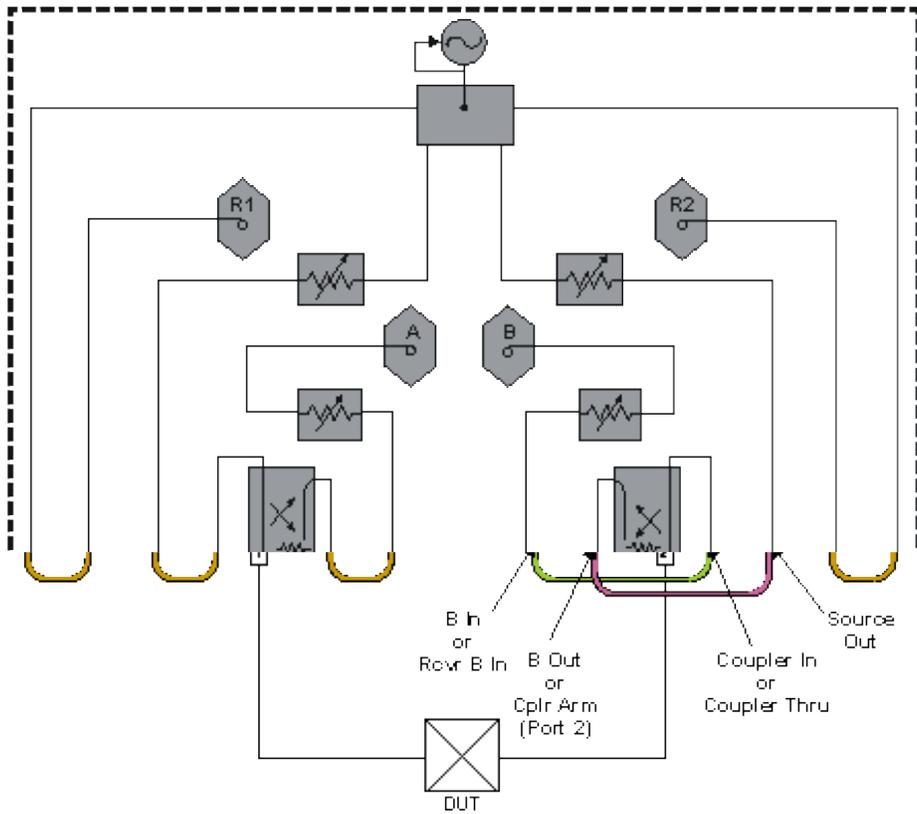
Reverse the directional coupler

Another method to improve dynamic range is to reverse the signal path in the test-port coupler and bypass the loss typically associated with the coupled arm.

As shown in the following graphic, the signal is applied to Port 2. The signal bypasses the coupled arm via the jumper cable connected to the Coupler Thru (or Coupler In) and the Receiver B In (or B In) ports.

When making measurements in reverse direction, the system dynamic range is degraded by 15 dB.

Note: Your analyzer's block diagram may contain different components than shown below. To see the block diagram for your PNA model, see the bottom of the [specs/data sheet](#).



See Also

[Front-panel Jumpers](#) (image)

[Using the Front Panel Jumpers](#)

[Specifications](#)

Last Modified:

- 5-Jan-2011 Removed old E835x std reference
- 5-Aug-2008 Combined several topics

Number of Points

A data point is a sample of data representing a measurement at a single stimulus value. You can specify the number of data points that the PNA measures across a sweep. (A "sweep" is a series of consecutive data point measurements, taken over a sequence of stimulus values.)

The PNA sweep time changes proportionally with the number of points. However, the overall measurement cycle time does not. See [Technical Specifications](#) for more information on how the number of points, and other settings, affect the sweep time.

How to change the number of data points

Select a number or click Custom to invoke a [dialog box](#)

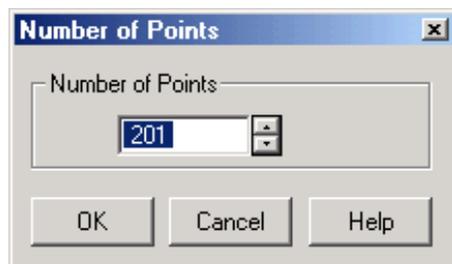
Using front-panel HARDKEY [softkey] buttons

1. Press **SWEEP**
2. then **[Number of Points]**

Using a mouse with PNA Menu

1. Click **Stimulus**
2. then **Sweep**
3. then **Number of Points**

← Programming Commands →



Number of Points dialog box help

Specifies the number of data points that the analyzer gathers during a measurement sweep. You can specify any number from **1** to **100,001**. The default value is 201.

Note: Some [measurement classes](#) may have different maximum points limitations.

Two data points are required for [Time Domain](#).

Tips:

- To achieve the greatest trace resolution, use the maximum number of data points.
- For faster throughput use the smallest number of data points that will give you acceptable resolution.
- To find an optimized number of points, look for a value where there is not a significant difference in the measurement when you increase the number of points.
- To ensure an accurate measurement calibration, perform the calibration with the same number of points that will be used for the measurement.

Last modified:

4-Dec-2013	Increased points max
6-Apr-2012	Removed speed note
2-Mar-2009	Increased max
3-Sep-2008	Removed legacy content
14-Dec-2007	Decreased min to 1
21-Jun-2007	MX Increased maximum
9/12/06	Added link to programming commands

Phase Measurement Accuracy

You can increase the accuracy of phase measurements by using the following PNA features

- [Electrical Delay](#)
- [Phase Offset](#)
- [Spacing Between Frequency Points \(Aliasing\)](#)

See Also

[Port Extensions](#)

[Comparing the PNA Delay Functions](#)

[Phase Control](#)

[Phase Coherent Measurements](#)

[Learn more about Phase measurements](#)

Electrical Delay

Electrical delay is a mathematical function that simulates a variable length of lossless transmission line.

Use the electrical delay feature to compensate for the linear phase shift through a device. This feature allows you to look at only the [deviation from linear phase](#) of the device.

You can set the electrical delay independently for each measurement trace.

How to set Electrical Delay

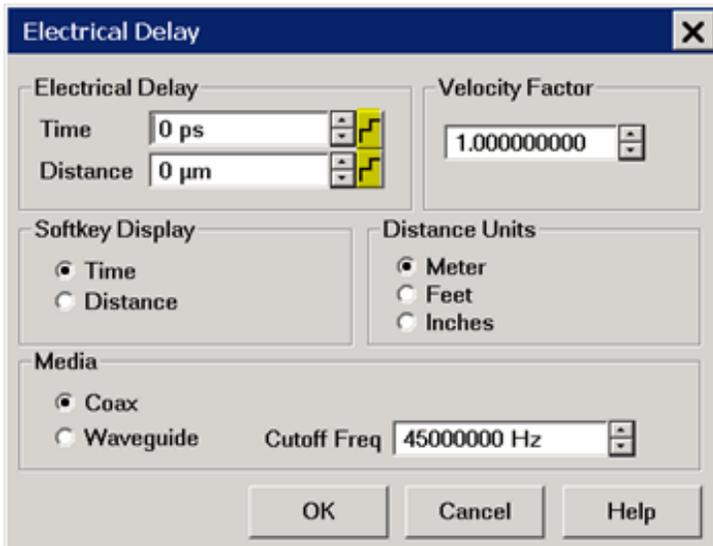
Using front-panel HARDKEY [softkey] buttons

1. Press **SCALE**
2. then **[Electrical Delay]**

Using a mouse with PNA Menus

1. Click **Response**
2. then **Scale**
3. then **Electrical Delay**

[Programming Commands](#)



Electrical Delay dialog box help

Electrical Delay Specifies the value of delay added or removed, in Time or Distance. This compensates for the linear phase shift through a device. You can set the electrical delay independently for each measurement trace.

Click the Step  icon next to either Time or Distance to start the [Step Size dialog](#).

Velocity Factor Specifies the velocity factor that applies to the medium of the device that was inserted after the measurement calibration. The value for a polyethylene dielectric cable is 0.66 and 0.7 for PTFE dielectric. 1.0 corresponds to the speed of light in a vacuum.

Velocity factor can also be set from the [Port Extensions](#) dialog and [Time Domain Distance Marker Settings](#).

Softkey Display Allows you to enter delay in either Time or Distance using the softkeys and [Active Entry toolbar](#).

Distance Units Select from Meters, Inches, or Feet. The step size will not change automatically when this value is changed. [Learn more about Step Size](#).

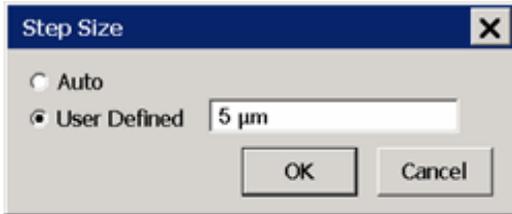
Media

Coax Select if the added length is coax. Also specify the velocity factor of the coax.

Waveguide Select if the added length is waveguide. Also specify the low frequency cutoff of the waveguide.

Cutoff Freq Low frequency cutoff of the waveguide.

Learn about [Electrical Delay](#) (scroll up)



Step Size dialog box help

Changes the step size that occurs when the Time or Distance up/down arrows are pressed on the Electrical Delay dialog.

Auto Step Size is set to the default value.

User Defined Enter a step size value, then click OK.

This value remains the same when the units are changed. For example if a step size of 12 is entered on this dialog, then you change the units from Inches to Feet, the step size of 12 inches becomes 12 feet, not 1 feet. Therefore, change the units first, then set the step size.

Phase Offset

Phase offset mathematically adjusts the phase measurement by a specified amount, up to 360°. Use this feature in the following ways:

- **Improve the display of a phase measurement.** This is similar to the way you would change the reference level in an amplitude measurement. Change the phase response to center or align the response on the screen.
- **Emulate a projected phase shift in your measurement.** For example, if you know that you need to add a cable and that the length of that cable will add a certain phase shift to your measurement, you can use phase offset to add that amount and simulate the complete device measurement.

How to set Phase Offset

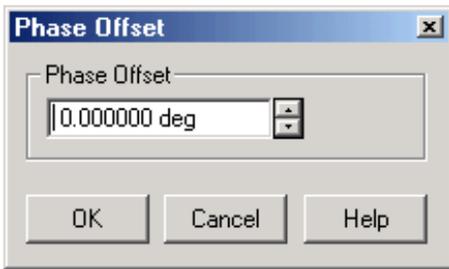
Using front-panel HARDKEY [softkey] buttons

1. Press **SCALE**
2. then **[Phase Offset]**

Using a mouse with PNA Menus

1. Click **Response**
2. then **Scale**
3. then **Phase Offset**

◀ Programming Commands ▶



Phase Offset dialog box help

Phase Offset Type a value or use the up and down arrows to select any value up to 360 degrees.

Learn about [Phase Offset](#) (scroll up)

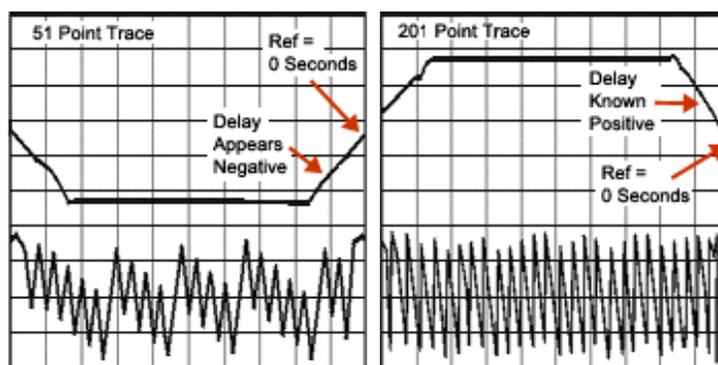
Spacing Between Frequency Points (Aliasing)

The analyzer samples data at discrete frequency points, then connects the points, creating a trace on the screen.

If the phase shift through a device is $>180^\circ$ between adjacent frequency points, the display can look like the phase slope is reversed. This is because the data is undersampled and aliasing is occurring.

If you are measuring group delay and the slope of the phase is reversed, then the group delay will change sign. For example, the following graphic shows a measurement of a SAW bandpass filter.

- The left measurement has 51 points and indicates the group delay is negative, which is a physical impossibility. That is, the response is below 0 seconds reference line.
- The right measurement shows an increase to 201 points which indicates the group delay is positive. That is, the response is above the 0 seconds reference line.



Tip: To check if aliasing might be occurring in a measurement, either [increase the number of points](#) or [reduce the frequency span](#).

Last modified:

12-Feb-2009	Added Step Size and Distance
3-Sep-2008	Removed legacy content
Sept.12, 2006	Added link to programming commands

Phase-Coherent Measurements

Coherent phase means that the measurement does NOT have random phase. Coherent-phase measurements require the comparison of phase between two sources that are sweeping simultaneously. It also requires that two receivers are measuring the two sources simultaneously.

This capability is available in the PNA-X using several applications:

- [iTMSA](#) (Opt 460) provides phase-coherent Balanced measurements.
- [Phase Control](#) (Opt 088) provides phase-coherent CW or Phase Sweep measurements.
- **R/R measurements** can provide simple phase-coherent measurements.

How to make phase-coherent R/R (R over R) measurements

To make a phase-coherent R/R measurements, both sources must be ON simultaneously, and the receiver measurements must be made on the same sweep. On a 4-port PNA-X, the two sources are NOT available at all PNA ports simultaneously. It is important to [learn these restrictions](#). Also, both receiver measurements must be made on the same sweep.

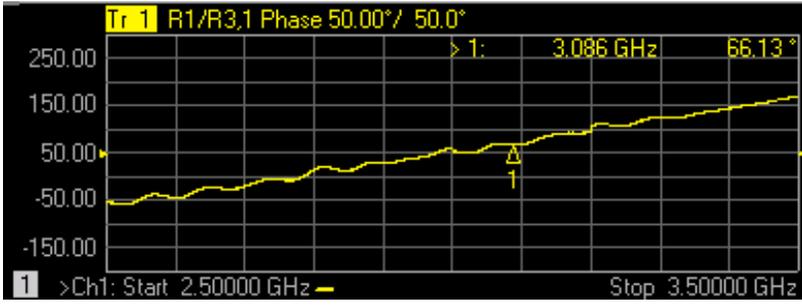
1. Create a ratioed receiver measurement using two reference receivers. [Learn how](#). For example, you might create an R1/R3 measurement, specifying the source port as either 1 or 3.



2. The source port that is selected above is turned on automatically. The other source port (port 3 in this case) must be turned ON manually using the [Power and Attenuators dialog](#).

Name	State
Port 1	Auto
Port 2	Auto
Port 3	ON
Port 4	Auto

3. Select a phase format. [Learn how](#).



Last Modified:

8-Apr-2011 Edited for accuracy
11-Nov-2010 MX New topic

Electrically-Long Device Measurements

A signal coming out of a device under test may not be exactly the same frequency as the signal going in to a device at a given instant in time. This can sometimes lead to inaccurate measurement results. You can choose between two techniques to eliminate this situation and increase measurement accuracy.

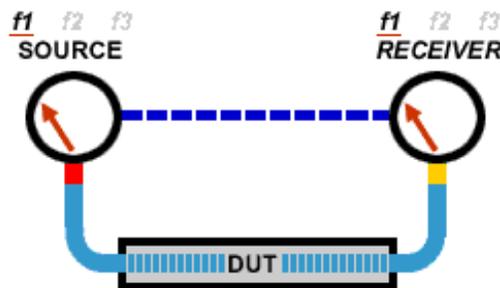
- [Why Device Delay May Create Inaccurate Results](#)
- [Solutions to Increase Measurement Accuracy](#)
 - [Slow the Sweep Speed](#)
 - [Add Electrical Length to the R Channel](#)

Other topics about Optimizing Measurements

Why Device Delay May Create Inaccurate Results

The following graphic shows an example of this situation:

- In the network analyzer, the source and receiver are **phase** locked together and sweep simultaneously through a span of frequencies.
- The signal flow through the Device Under Test (DUT) is shown as different colors for different frequencies.
- You can see as a stimulus frequency travels through the **DUT**, the analyzer tunes to a new frequency **just before** the signal arrives at the receiver. This causes inaccurate measurement results.



If the analyzer is measuring a long cable, the signal frequency at the end of the cable will lag behind the network analyzer source frequency. If the frequency shift is appreciable compared to the network analyzer's IF **detection** bandwidth (typically a few kHz), then the measured result will be in error by the rolloff of the IF filter.

Note: There is no fixed electrical length of a device where this becomes an issue. This is because there are many variables that lead to measurement speed. When high measurement accuracy is critical, lower the sweep speed until measurement results no longer change.

Solutions to Increase Measurement Accuracy

Choose from the following methods to compensate for the time delay of an electrically long device.

Slow the Sweep Speed

The following methods will slow the sweep speed.

- [Increase the Sweep Time](#)
- [Increase the Number of Points](#)
- [Use Stepped Sweep](#)
- [Set Dwell Time](#)

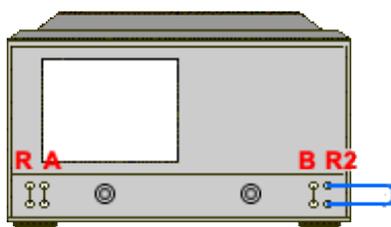
Add Electrical Length to the R Channel

Note: This method applies to PNA models with front panel loops.

Instead of slowing the sweep, you can compensate for the electrical length of a cable or fixture.

- Remove the R-channel jumper on the front panel of the analyzer.
- Replace the jumper with a cable of about the same length as the device under test.
 - Add the cable on the R1 channel for S_{11} and S_{21} measurements.
 - Add the cable on the R2 channels for S_{22} and S_{12} measurements.
- Set the analyzer for a fast sweep.

Configuration for S_{22} and S_{12} Measurements



This method balances the delays in the reference and test paths, so that the network analyzer's ratioed transmission measurement does not have a frequency-shift error.

Note: This method works well if the delay is in a cable or fixture. For devices with long delays, this method is only suitable for uncalibrated measurements.

Reflection Accuracy on Low-Loss 2-Port Devices

To make accurate reflection measurements that have a 1-port calibration, you should terminate the unmeasured port.

- [Why Terminate the Unmeasured Port](#)
- [How to Terminate the Unmeasured Port](#)
- [Resulting Measurement Uncertainty](#)

Other topics about Optimizing Measurements

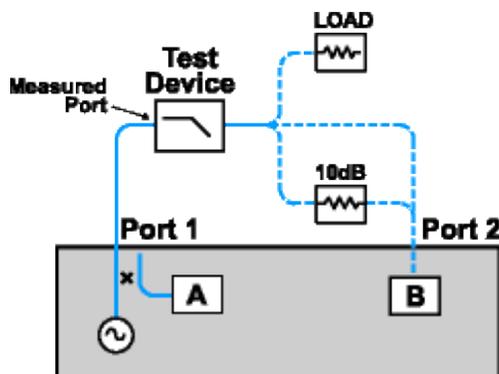
Why Terminate the Unmeasured Port

A 2-port calibration corrects for all 12 twelve error terms. A 1-port calibration corrects for directivity, source match and frequency response, but not load match. Therefore, for highest accuracy, you must make the load match error as small as possible. This especially applies for low-loss, bi-directional devices such as filter passbands and cables. You do not need to be concerned with load match when you are measuring a device with high reverse isolation, such as an amplifier.

How to Terminate the Unmeasured Port

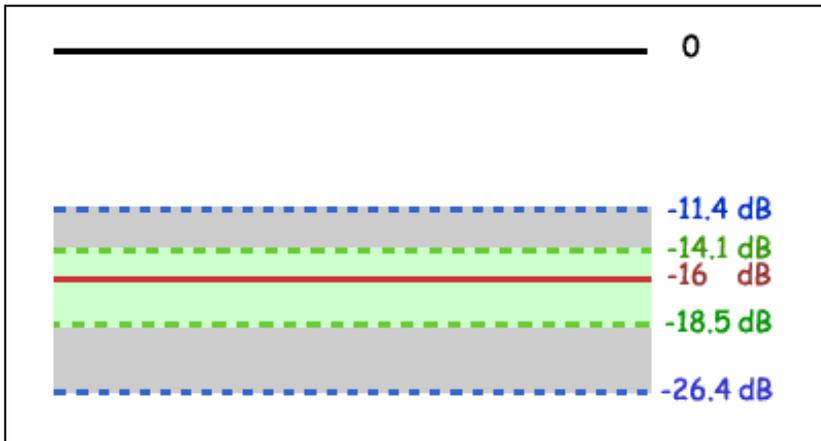
Use one of the following methods:

- Connect a high-quality termination load (from a calibration kit, for example) to the unmeasured port of your device. This technique yields measurement accuracy close to that of a Full SOLT 2-port calibration.
- Connect the unmeasured port of your device directly to the analyzer, inserting a 10 dB precision attenuator between the device output and the analyzer. This improves the effective load match of the analyzer by approximately twice the value of the attenuator, or 20 dB.



Resulting Measurement Uncertainty

The following graph illustrates the measurement uncertainty that results from terminating **with** and **without** a precision 10 dB attenuator on the output of the test device.



Legend

- Filter Reflection
- Uncertainty **with** attenuator
- Uncertainty **without** attenuator

The calculations below show how adding a high-quality 10 dB attenuator improves the load match of the analyzer.

Note: The corresponding linear value is shown in parentheses.

Network Analyzer:

$$\text{Load match (NA}_{LM}) = 18 \text{ dB } (.126)$$

$$\text{Directivity (NA}_{D}) = 40 \text{ dB } (.010)$$

Filter:

$$\text{Insertion loss (F}_{IL}) = 1 \text{ dB } (.891)$$

$$\text{Return loss (F}_{RL}) = 16 \text{ dB } (.158)$$

Attenuator:

$$\text{Insertion loss (A}_{\text{IL}}) = 10 \text{ dB (.316)}$$

$$\text{SWR (A}_{\text{SWR}}) = 1.05 (.024)$$

$$32.26 \text{ dB Return Loss}$$

Calculations:

	Without Attenuator	With Attenuator
ρ_{NA}	$= (\text{F}_{\text{IL}}) * (\text{N}_{\text{ALM}}) * (\text{F}_{\text{IL}})$ $= (.891) * (.126) * (.891)$ $= .100$	$= (\text{F}_{\text{IL}}) * (\text{A}_{\text{IL}}) * (\text{N}_{\text{ALM}}) * (\text{A}_{\text{IL}}) * (\text{F}_{\text{IL}})$ $= (.891) * (.316) * (.126) * (.316) * (.891)$ $= .010$
$\rho_{\text{Attenuator}}$	NA	$= (\text{F}_{\text{IL}}) * (\text{A}_{\text{SWR}}) * (\text{F}_{\text{IL}})$ $= (.891) * (.024) * (.891)$ $= .019$
Worst Case Error (EWC)	$= \rho_{\text{NA}}$ $= .1$	$= \rho_{\text{NA}} + \rho_{\text{Attn.}}$ $= .01 + .019$ $= .029$

Uncertainty Adds	$= -20\log(\text{F}_{\text{RL}}) + (\text{EWC}) + (\text{NAD})$ $= -20\log(.158) + (.100) + (.010)$ $= \mathbf{11.4 \text{ dB}}$	$= -20\log(\text{F}_{\text{RL}}) + (\text{EWC}) + (\text{NAD})$ $= -20\log(.158) + (.029) + (.010)$ $= \mathbf{14.1 \text{ dB}}$
-------------------------	--	--

Uncertainty Subtracts	$= -20\log(\text{F}_{\text{RL}}) - (\text{EWC}) - (\text{NAD})$ $= -20\log(.158) - (.100) - (.010)$ $= \mathbf{26.4 \text{ dB}}$	$= -20\log(\text{F}_{\text{RL}}) - (\text{EWC}) - (\text{NAD})$ $= -20\log(.158) - (.029) - (.010)$ $= \mathbf{18.5 \text{ dB}}$
------------------------------	--	--

Last Modified:

10-Mar-2010 Fixed Aswr- thanks Alex!

Measurement Stability

There are several situations that can cause unstable measurements. To ensure that you are making repeatable measurements, you can use various methods to create a stable measurement environment.

- [Frequency Drift](#)
- [Temperature Drift](#)
- [Inaccurate Measurement Calibrations](#)
- [Device Connections](#)

Other topics about Optimizing Measurements

Frequency Drift

The analyzer frequency accuracy is based on an internal 10 MHz frequency oscillator. See [Technical Specifications](#) for stability and aging specifications.

If your measurement application requires better frequency accuracy and stability, you can override the internal frequency standard and provide your own high-stability external frequency source through the 10 MHz Reference [Input connector on the rear panel](#).

Temperature Drift

Thermal expansion and contraction changes the electrical characteristics of the following components:

- Devices within the analyzer
- Calibration kit standards
- Test devices
- Cables
- Adapters

To reduce the effects of temperature drift on your measurements, do the following.

- Switch on the analyzer 1/2 hour before performing a measurement calibration or making a device measurement.
- One hour before you perform a measurement calibration, open the case of the calibration kit and take the

standards out of the protective foam.

- Use a temperature-controlled environment. All specifications and characteristics apply over a 25 °C \pm 5 °C range (unless otherwise stated).
- Ensure the temperature stability of the calibration kit devices.
- Avoid handling the calibration kit devices unnecessarily during the calibration procedure.
- Ensure the ambient temperature is \pm 1°C of the measurement calibration temperature.

Inaccurate Measurement Calibrations

If a measurement calibration is inaccurate, you will not measure the true response of a device under test. To ensure that your calibration is accurate, you should consider the following practices:

- Perform a measurement calibration at the points where you connect the device under test, that is, the reference plane.
- If you insert any additional accessory (cable, adapter, attenuator) to the test setup after you have performed a measurement calibration, use the port extensions function to compensate for the added electrical length and delay.
- Use calibration standards that match the definitions used in the calibration process.
- Inspect, clean, and gage connectors. See [Connector Care](#).

See [Accurate Measurement Calibrations](#) for more detailed information.

Device Connections

Good connections are necessary for repeatable measurements. To help make good connections, do the following:

- Inspect and clean the connectors for all of the components in the measurement setup.
- Use proper connection techniques.
- Avoid moving the cables during a measurement.

Noise Reduction Techniques

Random electrical noise which shows up in the analyzer receiver chain can reduce measurement accuracy. The following PNA functions help reduce trace noise and the noise floor which can lead to better dynamic range and more accurate measurements.

Note: The trace noise in microwave PNAs becomes worse below 748 MHz and is especially obvious between 10 MHz and 45 MHz. See [Reduce IFBW](#).

- [Averaging](#)
- [IF Bandwidth](#)
- [Trace Smoothing](#)

See Also

[Group Delay](#)

[Increase Dynamic Range](#)

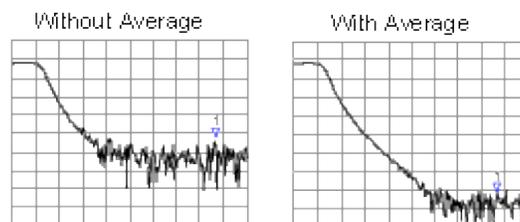
[PNA data processing map](#).

Other topics about Optimizing Measurements

Averaging

Averaging is a feature that reduces the effects of random noise on a measurement. The PNA computes **each data point** based on the average of several measurements. You determine the number of measurements by setting the Average factor. The higher the average factor, the greater the amount of noise reduction.

Effects of Sweep Average



Both **Averaging** and [IF Bandwidth](#) can be used for the same benefit of general noise reduction. For minimizing very low noise, Averaging is more effective than reducing IF bandwidth. Generally, Averaging takes slightly longer than IF bandwidth reduction to lower noise, especially if many averages are required. Also, changing the IF bandwidth after calibration results in [uncertain accuracy](#).

How to Set Averaging

Using front-panel HARDKEY [softkey] buttons

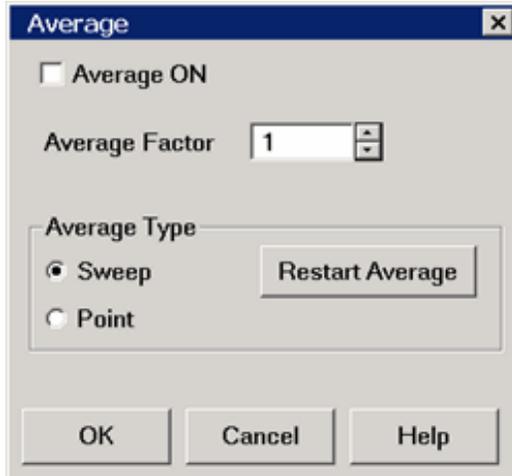
1. Press **AVG**
2. then [**Averaging**]

Using a mouse with PNA Menus

1. Click **Response**
2. then **Avg**
3. then **Average**

Programming Commands

Average dialog box help



Average ON Check to enable Averaging.

Average Factor Specifies the number of measurements that are averaged. Range of 1 to 65536 (2^{16}).

Average Type

Sweep Each data point is based on the average of the same data point measured over consecutive sweeps. When the number of sweeps = Average Factor, the averaging continues following the [Sweep Averaging formula](#).

(Sweep) Restart Begins a new set of measurements that are used for the average. Applies only to Sweep averaging - NOT Point.

Point Each data point is measured the number of times specified by the Average Factor, and then averaged, before going to the next data point.

- On subsequent sweeps, averaging is automatically restarted by measuring each data point again the number of times specified by the Average Factor.
- Because measurements occur quickly in the background, the Average Counter is NOT updated.
- Point averaging is NOT available in [Gain Compression](#), or [Noise Figure](#) Apps.

Notes

- An **Average Counter** appears on the screen when Sweep averaging is selected, displaying the number of sweeps that has been averaged. The effect on the signal trace can be viewed as the Average Factor increases. This can assist in the selection of the optimum number of sweep averages. The Average Counter is NOT updated for **Point** averaging.
- **Channel-wide scope-** Averaging is enabled and the factor is set for all measurements in a channel. The Average counter is displayed for each channel.
- **Calibration** - Because averaging is a mathematical process that occurs after the raw measurement is made, averaging can be turned ON before or after calibration without invalidating the error correction terms. If averaging is ON before calibration, the measurement of calibration standards are averaged measurements. More time is needed to perform the calibration, but there will be less noise in the resulting error correction terms. Subsequent corrected measurements will also have less noise error. In addition, noise is further reduced by turning Averaging ON after calibration. [See the PNA data processing map.](#)
- **PNA Triggering** is implemented separately from Averaging. For example, setting averaging factor to 4 has NO effect on the number of triggers that are required to achieve 4 sweeps or 4 data points.
- **Unratioed** measurements - Although averaging unratioed (single receiver) measurements is allowed, you may see unexpected results.
 - The noise floor does not drop when averaging unratioed measurements as on ratioed measurements.
 - Phase results may tend toward 0. This is because phase measurements are relative by nature. Measuring absolute phase with a single receiver appears random. Averaging random positive and negative numbers will tend toward 0.

Sweep Averaging Formula

$$\text{NewAvg} = (\text{NewData}/n) + [\text{OldAvg} * (n-1/n)]$$
 'where n = average factor'

From the formula, you can see that data from the first **n** sweeps continues to be included in the results of subsequent sweeps. Its effect is increasingly smaller but never diminishes to zero. For example, with $n = 5$, the average of the 5 sweeps is displayed. On the 6th sweep, you see $4/5$ the average of the first 5 sweeps plus $1/5$ the new sweep.

The effects of older data can be eliminated by clicking **Restart**.

[Learn more about Averaging](#) (scroll up)

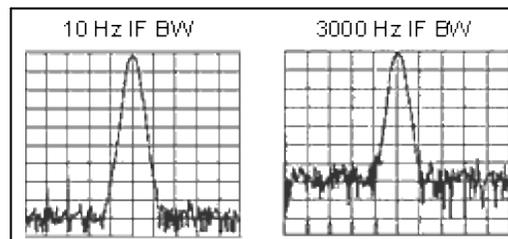
IF Bandwidth

The PNA converts the received signal from its source to a lower intermediate frequency (IF). The bandwidth of the IF bandpass filter is adjustable from 40 kHz (for most PNA models) down to a minimum of 1 Hz.

Reducing the IF receiver bandwidth reduces the effect of random noise on a measurement. Each tenfold reduction in IF bandwidth lowers the noise floor by 10 dB. However, narrower IF bandwidths cause longer sweep times.

- **Channel wide** - IF bandwidth can be set independently for each channel
- **Segment sweep** - IF bandwidth can be set independently for each segment of segment sweep.
- **Calibration** - Changing the IF bandwidth after calibration will cause a ['C-delta' correction level](#), which means that calibration accuracy is uncertain.

Effect of Reducing IF Bandwidth



How to set IF Bandwidth

Using front-panel HARDKEY [softkey] buttons

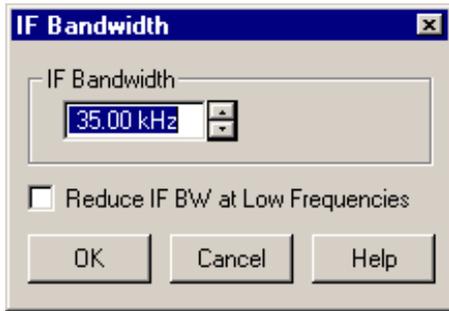
1. Press **Avg**
2. then **[IF Bandwidth]**

Using a mouse with PNA Menus

1. Click **Response**
2. then **Avg**
3. then **IF Bandwidth**

[Programming Commands](#)

[IF Bandwidth dialog box help](#)



IF Bandwidth Specifies the IF (receiver) bandwidth. The value of IF bandwidth is selected by scrolling through the values available in the IF bandwidth text box. The IF BW is set independently for each channel.

The list of selectable IF Bandwidths is different depending on PNA model.

The following values are common to all models:

- **1 | 2 | 3 | 5 | 7 | 10 | 15 | 20 | 30 | 50 | 70 | 100 | 150 | 200 | 300 | 500 | 700 | 1k | 1.5k | 2k | 3k | 5k | 7k | 10k | 15k | 20k | 30k**

In addition, the following values are PNA Model specific:

5230A/C Opts 020, 025, 120, 125, 140, 145, 146, 240, 245, 146

- **50k | 70k | 100k | 150k | 200k | 280k | 360k | 600k**

N5230A/C Opts 220, 225, 420, 425, 520, 525:

- **50k | 70k | 100k | 150k | 200k | 250k**

E836x models:

- **35k | 40k**

PNA-X and N522x:

- **50k | 70k | 100k | 150k | 200k | 280k | 360k | 600k | 1M | 1.5M | 2M | 3M | 5M | 7M | 10M | 15M**

- For the highlighted IFBW settings (1 MHz and above):

- 1 MHz to **5 MHz** settings are available ONLY with [DSP version](#) 4.0 and above.
- 7 MHz to **15 MHz** settings are available ONLY with [DSP version](#) 5.0 and above.
- These settings are used to make [wideband pulsed](#) measurements. They do NOT provide faster

sweep speeds for non-pulsed measurements.

- A slight shift (1dB or more) in Log Mag traces may be seen when switching in and out of these bandwidths.
- Available in [Step sweep](#) mode only - NOT available in Analog sweep.

Reduce IF BW at Low Frequencies

On PNA models with a maximum frequency of 20 GHz and higher, the trace noise becomes worse below about 400 MHz. This is especially obvious between 10 MHz and 45 MHz and also when Time Domain is ON. [See PNA models / maximum frequencies.](#)

When this box is checked, the PNA uses a smaller IF Bandwidth than the selected value at frequencies indicated below.

This setting:

- can be made for each channel.
- is ON (checked) by default.
- also applies to [segment sweep](#).

Use the following calculations to determine the actual IF Bandwidth that is used. If the result is NOT a selectable IF BW, the next higher selectable value is used.

In the following table and example, the next band starts at .01 Hz above the Stop Frequency. This is indicated with '+'.

Stop Frequency	PNA Model						
	N5241A N5242A	N5221A N5222A	N5244A N5245A	N5224A N5225A	N5247A	N5227A	N5234A N5235A
14 MHz	0.05	0.05	0.005	0.005	0.005	0.005	0.025
19 MHz	0.05	0.05	0.005	0.005	0.005	0.005	0.025
27 MHz	0.05	0.05	0.01	0.005	0.01	0.005	0.025
38 MHz	0.05	0.05	0.01	0.005	0.01	0.005	0.025
53 MHz	0.05	0.05	0.015	0.005	0.015	0.005	0.025
75 MHz	0.1	0.1	0.025	0.025	0.025	0.025	0.025
105 MHz	0.1	0.1	0.05	0.025	0.05	0.025	0.025
146 MHz	0.14	0.1	0.1	0.025	0.1	0.025	0.025
175 MHz	0.14	0.1	0.15	0.025	0.15	0.025	0.025
205 MHz	0.29	0.29	0.15	0.15	0.15	0.15	0.025
250 MHz	0.5	0.29	0.25	0.15	0.25	0.15	0.025

396 MHz	1	1	0.5	0.5	0.5	0.5	1
396+ MHz and above	1	1	1	1	1	1	1

Example:

On a N5224A, the selected IF BW is 30 KHz.

With **Reduce IF BW at Low Frequencies** checked, the actual IF Bandwidths used are:

- From 53+ MHz to **175 MHz**: $30,000\text{Hz} * .025 = 750 \text{ Hz}$ (next higher selectable value: **1 kHz**.)
- From 175+ MHz to **250 MHz**: $30,000\text{Hz} * .15 = 4.5 \text{ kHz}$ (next higher selectable value: **5 kHz**.)
- From 250+ MHz to 396 MHz: $30,000\text{Hz} * .5 = 15 \text{ kHz}$
- From 396+ MHz to stop sweep = 30 kHz

OK Selects the IF BW value shown in the text box.

[Learn about IF Bandwidth](#) (scroll up)

Trace Smoothing

Trace smoothing averages a number of **adjacent** data points to smooth the displayed trace. The number of adjacent data points that get averaged together is also known as the smoothing aperture. You can specify aperture as either the number of data points or the percentage of the x-axis span.

Trace Smoothing reduces the peak-to-peak noise values on broadband measured data. It smooths trace noise and does not increase measurement time significantly.

Because Trace Smoothing follows Format in the PNA data processing map, the formatted data is smoothed. Smoothing is automatically turned off if the format is Polar or Smith Chart.

[Learn more about Data Format Types.](#)

[See the PNA data processing map.](#)

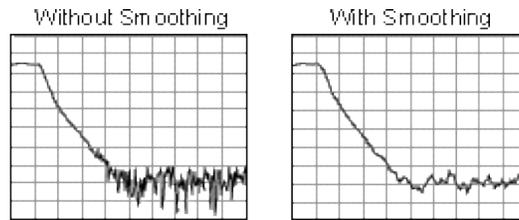
Tips:

- Start with a high number of display points and reduce until you are confident that the trace is not giving misleading results.
- Do not use smoothing for high-resonance devices, or devices with wide trace variations. It may introduce

misleading information.

- Smoothing is set independently for each trace.

Effects of Smoothing on a Trace



How to set Trace Smoothing

Using front-panel HARDKEY [softkey] buttons

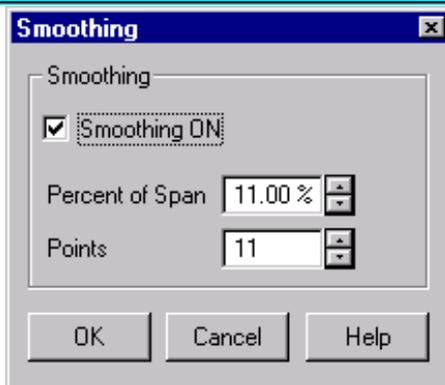
1. Press **Avg**
2. then [**Smoothing**]

Using a mouse with PNA Menus

1. Click **Response**
2. then **Avg**
3. then **Smoothing**

◀ Programming Commands ▶

Smoothing dialog box help



Smoothing ON When checked, applies smoothing to the displayed trace.

Percent of Span Specify percent of the swept stimulus span to smooth. For example, for a trace that contains 100 data points, and specify a percent of span = 11%, then the number of data points

that are averaged is 11.

Points Specify the number of adjacent data points to average.

[Learn about Trace Smoothing](#) (scroll up)

Last modified:

7-Mar-2012	Restricted highlighted IFBW notes
30-Dec-2011	Several edits to Averaging
13-Oct-2011	Added wider IFBWs
15-Feb-2011	Removed point NOT on C models and Noise Fig.
22-Dec-2009	Added avg formula
10-Aug-2009	Point Avg Not available on GCA or NF Apps.
8-Jun-2009	Point averaging not available on C models.
6-Apr-2009	Replaced N5242A with PNA-X
12-Mar-2009	Edit IFBW for WB pulse
9-Oct-2008	Updated for Point Average (8.33)
3-Sep-2008	Removed legacy content

Crosstalk

Crosstalk is energy leakage between analyzer signal paths. This can be a problem with high-loss transmission measurements. Although the [crosstalk specification](#) of the PNA is exceptional, you can reduce the effects of crosstalk by doing the following:

- [Set the Sweep to Alternate](#)
- [Perform an Isolation Calibration](#)

Other topics about Optimizing Measurements

Set the Sweep to Alternate

This selection is longer available from the user interface. [Learn more.](#)

Perform an Isolation Calibration

For transmission measurements, a response and isolation measurement calibration helps reduce crosstalk because the analyzer measures and then subtracts the leakage signal during the measurement calibration. The calibration improves isolation so that it is limited only by the noise floor.

Note: Isolation is never performed on a Smart (Guided) Calibration. [Learn more.](#)

Generally, the isolation error falls below the noise floor. So when you are performing an isolation calibration you should use a noise reduction technique such as sweep averages or reducing the IF bandwidth.

Last Modified:

3-Mar-2010 Edited alternate sweep text

Effects of Accessories

Accessories in a configuration may affect the results of a device measurement. You can choose between two analyzer features that reduce the effects of accessories

- [Power Slope to Compensate for Cable Loss](#)
- [Gating to Selectively Remove Responses](#)

Other topics about Optimizing Measurements

Power Slope to Compensate for Cable Loss

If you have a long cable or other accessory in a measurement configuration where a power loss occurs over frequency, apply the power slope function. This function increases the analyzer source power by a rate that you define (dB/GHz).

1. In the **Channel** menu, click **Power**.
2. If the slope function is not already switched on, click the **Slope** check box.
3. In the **dB/GHz** box, enter the rate that you want the source power to increase over the frequency sweep. Click **OK**.

Gating to Selectively Remove Responses

Gating is a feature in the time domain (option 010) that allows the analyzer to mathematically remove responses. You can set the gate for either a reflection or transmission response, but you will see different results.

- **Gating a reflection response** isolates a desired response (such as a filter's return loss), from unwanted responses (such as adapter reflections or connector mismatches).
- **Gating a transmission response** isolates a specific path in a multipath device that has long electrical lengths.

See [Time Domain Gating](#) for more information.

Achieve Fastest Sweep

You can achieve the fastest measurement sweep by adjusting the following:

- [Sweep Settings](#)
- [Noise Reduction Settings](#)
- [Measurement Calibration Choice](#)
- [Unnecessary Functions](#)

Other topics about Optimizing Measurements

Sweep Settings

Consider changing each of the following settings as suggested.

- [Frequency Span](#) - Measure only the frequencies that are necessary for your device.
- [Segment Sweep](#) - Use segments to focus test data only where you need it.
- [Switch Off Stepped Sweep](#) - Use linear swept mode to minimize sweep time when possible.
- [Auto Sweep Time](#) - Use this default to sweep as quickly as possible for the current settings.
- [Number of Points](#) - Use the minimum number of points required for the measurement.

For more information on how number of points and other settings affect sweep cycle time, see [Technical Specifications](#).

Noise Reduction Settings

Using a combination of these settings, you can decrease the sweep time while still achieving an acceptable measurement.

- [IF Bandwidth](#). Use the widest IF bandwidth that will produce acceptable trace noise and [dynamic range](#).
- [Average](#). Reduce the average factor, or switch Average off.

Measurement Calibration Choice

Choose the appropriate type of calibration for the required level of accuracy.

When full 2-port error correction is applied, the PNA takes both forward and reverse sweeps to gather all 12 error correction terms. This occurs even with a single S11 measurement displayed. All displayed measurements are updated as the second sweep is performed. Both sweeps are performed using the specified sweep time.

When calibrating greater than 2 ports, the following formula is used to determine the number of sweeps required:

- $N * (N-1)$ where N = the number of ports.

When full 3-port calibration is applied, 6 sweeps are required; forward and reverse for each port pair. With full 4-port correction, 12 sweeps are required, and so forth.

To limit the measurement time, perform ONLY the level of calibration that your measurements require. For example, if making only an S11 measurement, perform a 1-port calibration on that port.

Sweep speed is about the same for uncorrected measurements and measurements done using a response calibration, or one-port calibration. For more information see [Select a Calibration](#).

Unnecessary Functions

The analyzer must update information for all active functions. To achieve an additional increase in sweep speed, switch off all of the analyzer functions that are not necessary for your measurement application.

- [Delete Unwanted Traces](#)
- [Switch Off Unwanted Markers](#)
- [Switch Off Smoothing](#)
- [Switch Off Limit Testing](#)
- [Switch Off Math Functions](#)

Analyzer sweep speed is dependent on various measurement settings. Experiment with the settings to get the fastest sweep and the measurement results that you need.

Switch Between Multiple Measurements

If you need to make multiple measurements to characterize a device, you can use various methods to increase throughput. Experiment with these methods to find what is best for your measurement application needs.

- [Set Up Measurements for Increased Throughput](#)
 - [Arrange Measurements in Sets](#)
 - [Use Segment Sweep](#)
 - [Trigger Measurements Selectively](#)
- [Automate Changes Between Measurements](#)
- [Recall Measurements Quickly](#)

Other topics about Optimizing Measurements

Set Up Measurements for Increased Throughput

To achieve optimum throughput of devices that require multiple measurements, it is helpful to know the operation of the PNA. This knowledge allows you to set up the measurement scenarios that are best for your applications.

[Learn more about Traces, Channels, and Windows on the PNA](#)

Arrange Measurements in Sets

If you arrange measurements to keep the complete set of device measurements in one instrument state, you can save them so that you can later recall a number of measurements with one recall function.

See [Pre-configured Measurement Setups](#) for more information.

Use Segment Sweep

Segment sweep is helpful if you need to change the following settings to characterize a device under test.

- Frequency Range

- Power Level
- IF Bandwidth
- Number of Points

The segment sweep allows you to define a set of frequency ranges that have independent attributes. This allows you to use one measurement sweep to measure a device that has varying characteristics.

See [Segment Sweep](#) for more information.

Trigger Measurements Selectively

You can use the measurement trigger to make measurements as follows:

- Continuously update only the measurements that have rapidly changing data.
- Occasionally update measurements that have infrequently changing data.

For example, if you had four channels set up as follows:

- Two channels measuring the data that is used to tune a filter
- Two channels measuring the data for the out-of-band responses of the filter

You would want to constantly monitor only the measurement data that you use for tuning the filter. If you continuously update all of the channels, this could slow the response of the analyzer so that you would not be able to tune the filter as effectively.

Note: You must either trigger the infrequent measurement manually or with remote interface commands.

To trigger measurements selectively:

This procedure shows you how to set up two different measurements with the following behavior:

- Channel 1 measurement will continuously update the data.
- Channel 2 measurement will occasionally update the data.

1. In the **Windows** menu, click **Meas Setups, Setup D**.

Set Up a Measurement Trigger for Continuous Updates

2. In the **Sweep** menu, click **Trigger, Trigger...**
3. Under **Trigger Source**, click **Internal**.
4. Under **Channel Trigger State**, select **Channel 1**, and click **Continuous**.

Set Up a Measurement Trigger for Occasional Updates

5. Under **Channel Trigger State**, select **Channel 2**, and click **Single, OK**.
 - o If you want the analyzer to trigger more than a single sweep, click the **Enable Groups** check box and enter the number of sweeps.
6. In the **System** menu, click **Keys, Trigger**.

Update the Measurement

7. Click on the lower window to make Channel 2 the **active channel**.
8. On the active entry toolbar, click the type of trigger you set up.
 - o Click **Single** if you set up the analyzer for a single sweep per trigger.
 - o Click **Groups** if you set up the multiple sweeps per trigger.

Note: A trace must be active for you to initiate a trigger for that measurement.

Automate Changes Between Measurements

If there are slight differences between the various measurements that you need to characterize a device, you may find that it is faster to change the measurement settings using programming.

Recall Measurements Quickly

The most efficient way to recall measurements is to recall them as a set of measurements (instrument state).

- It only takes a short time longer to recall an instrument state that includes multiple measurements, than it does to recall an instrument state with only one measurement.
- Each recall function has time associated with it. You can eliminate that time by setting up the measurements as a set so you can recall them as a set.

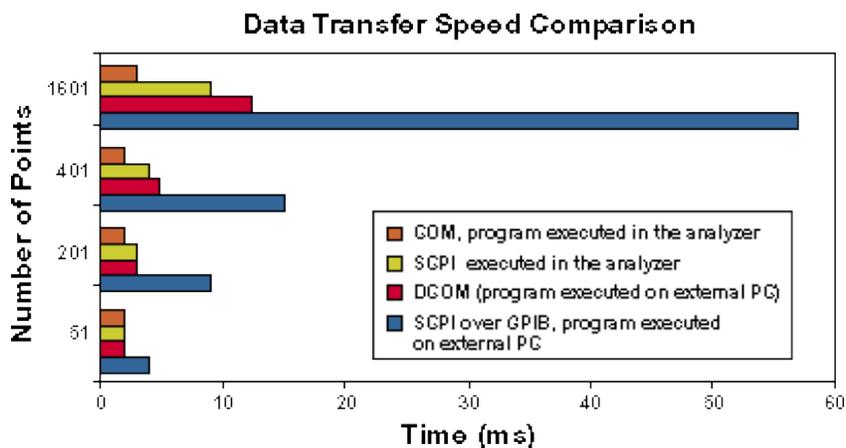
See [Save and Recall Files](#) for more information.

Data Transfer Speed

When testing devices remotely using COM or SCPI, the following techniques can be used to transfer data quickly between the PNA and remote computer, helping you achieve the best measurement throughput.

- Use [single sweep \(trigger\) mode](#) to ensure that a measurement is complete before starting a data transfer.
- **Transfer the minimum amount of data** needed. For example, a trace with a few points, using segment sweep rather than a full trace with many linearly spaced points. Also, use markers instead of trace transfers.
- **Choose the REAL data format** to provide the fastest transfer speed when using SCPI programs for automated applications.
- **Use SCPI over LAN** for applications that are automated with SCPI programs.
- **Use COM programs** to provide the fastest transfer speed when using an automated application. See [Data Transfer Time](#) specifications.

Note: The following data is obsolete, but still serves to illustrate the relative speed between COM and SCPI.



[Other topics about Optimizing Measurements](#)

Using Macros

Macros are executable programs that you write, load into the analyzer, and then run from the analyzer. You can have up to 25 macros set up to run on the analyzer.

- [How to Setup Macros](#)
- [How to Run Macros](#)
- [Macro Example](#)

How to Setup Macros

Using front-panel HARDKEY [softkey] buttons

1. Press **MACRO**
2. then **[Macro Setup]**

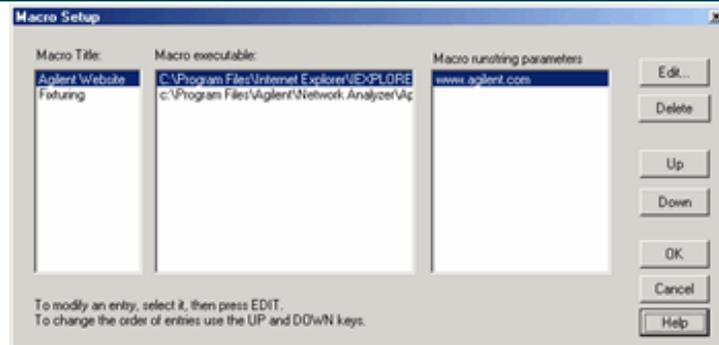
Using a mouse with PNA Menus

1. Click **Utility**
2. then **Macro**
3. then **Macro Setup**

In the Macro Setup dialog box:

1. Create an executable program and save it on the PNA hard drive. See [SCPI](#) or [COM](#) example programs in VBScript.
2. Use a mouse or the front-panel 'down-arrow' to select a blank line below the last entry. (There may be NO entry.)
3. Click **Edit** to start the [Edit Macro Setup](#) dialog.
4. In the **Macro Title** box, type a descriptive title for your macro.
5. Click **Browse**.
6. Change **Files of Type**.
7. Find and select your executable file. Change **Files of Type** if necessary.
8. Click **OK** on the Edit Macro Setup dialog.
9. Click **OK** on the Macro Setup dialog.
10. Press **MACRO** to run. It may be necessary to first Preset the PNA to see your macro in the menu.

Macro Setup dialog box help



Macro setup allows you to create up to 25 macros that can be launched from the PNA application.

An external keyboard is required to enter the Macro Title and the Run string parameters.

To add a Macro, use a mouse or the front-panel 'down arrow' (NOT the 'Down' key) to select a blank line. Then click **Edit**.

Macro Title Shows the titles that appear in the softkeys and menu when you press the Macro key. These titles are associated with the executable files and should be descriptive so you can easily identify them.

Macro Executable Lists the complete path to the executable file. To follow the example of launching the Agilent PNA Series Home Page, the path to the executable could be "C:/Program Files/Internet Explorer/iexplore.exe".

Macro Runstring Parameters Lists the parameters that get passed to the program that is referenced in the executable file. Again following the example of launching the PNA Series Home Page, you could assign the runstring parameters "http://www.agilent.com/find/pna".

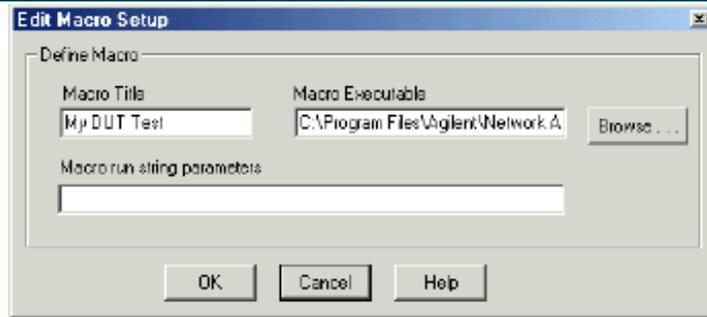
Edit Invokes the [Macro Edit dialog box](#).

Delete Deletes the selected macro.

Up Allows you to reorder the macros, moving the selected macro up one line. This order determines how they appear in the PNA Menu and in the softkeys and when you press the Macro front-panel key.

Down Moves the selection down one line in the list of macros.

Macro Edit dialog box help



Macro Title Add a title that appears in the softkeys and menu.

Macro Executable Set the complete path to the macro executable file. Click **Browse** to navigate to the macro executable file and establish the complete path to the file.

Macro run string parameters Optionally add parameters that are passed to the program referenced in the executable file.

[See Macro Setup dialog box](#)

How to Run Macros

Using front-panel HARDKEY [softkey] buttons

1. Press **MACRO**
2. then select the macro to run

Using a mouse with PNA Menus

1. Click **Utility**
2. then **Macro**
3. then select the macro to run

◀ Programming Commands ▶

Macro Example

The following is an example Visual Basic Scripting (vbs) program that you can copy, install, and run on your PNA.

Note: Print these instructions if viewing in the analyzer. This topic will be covered by the Macro Setup dialog box.

1. Copy the following code into a Notepad file.
2. Save the file on the analyzer hard drive in the **C:/Documents** folder. Name the file **FilterTest.vbs**
3. Close Notepad
4. [Setup the macro in the PNA](#)
5. [Run the macro](#)

```
'Start copying here
'This program creates a S21 measurement, with Bandwidth
'markers for testing a 175MHz Bandpass filter
'It is written in VBscript using COM commands

Set PNA = CreateObject("AgilentPNA835x.Application")
PNA.Preset
Set chan=PNA.activechannel
Set meas=PNA.activemeasurement
Set limts = meas.LimitTest
Set trce = PNA.ActiveNAWindow.ActiveTrace

meas.ChangeParameter "S21",1
chan.StartFrequency = 45e6
chan.StopFrequency = 500e6
trce.ReferencePosition = 8
PNA.TriggerSignal = 3

'Do Test
for t=1 to 5
call measure
call compare
next
msgbox("Done Testing")

sub measure
msgbox("Connect Device " & t & " and press OK")
PNA.ManualTrigger True
meas.SearchFilterBandwidth
end sub

sub compare
BW = meas.FilterBW
if bw>6.5e7 then msgbox("Failed BW: " & BW)
Loss = meas.FilterLoss
if loss>5 then msgbox("Failed Loss: " & Loss)
end sub
'End copying here
```

Last Modified:

3-May-2012	Removed c models
17-Feb-2009	Added 25 limit
4-Sep-2008	Removed legacy content

Select a Calibration Type

The following calibration types are available in the PNA.

Cal Type	Interface	Accuracy	Thru Methods allowed
TRL Family	Both	Very High	All except Unknown Thru
SOLT	Both	High	All
Enhanced Response	SmartCal	High	Defined Thru or Flush Thru
QSOLT (Quick SOLT)	SmartCal	Medium	Defined Thru or Flush Thru
1-Port Reflection	Both	High	Not Applicable
Open/Short Response	Unguided	Low	Not Applicable
Thru Response	Unguided	Low	Known Thru or Flush Thru

[Learn how to select a default Cal Type.](#)

Other Cal Types (Separate Topic)

- [Source and Receiver Power Cals](#)

[See other Calibration Topics](#)

TRL Family

Application: Used to accurately calibrate any pair of ports when calibration standards are not readily available.

Note: A [Delta Match Cal](#) may be required.

- [Learn more about TRL family cal](#)
- For more information on modifying standards, see [Calibration Standards](#).

Calibration Method: [SmartCal](#), [Unguided Calibration](#)

General Accuracy: Very High

Standards Required: THRU, REFLECT, LINE or similar combination

Systematic Errors Corrected:

- Directivity
 - Source match
 - Isolation ([see exceptions](#))
 - Load match
 - Frequency response transmission tracking
 - Frequency response reflection tracking
-

SOLT

Application: Used to accurately calibrate any number of ports.

General Accuracy: High

Calibration Method: [SmartCal](#), [Unguided Calibration](#), [ECal](#)

Standards Required: (SHORT, OPEN, LOAD, THRU) or ECal module

Systematic Errors Corrected (on all ports):

- Directivity
 - Source match
 - Isolation ([see exceptions](#))
 - Load match
 - Frequency response transmission tracking
 - Frequency response reflection tracking
-

Enhanced Response

Application: Used to calibrate two ports when only measurements in one direction (forward OR reverse) are required. Measurements are faster because a second sweep is NOT required.

- Reflection Standards (OPEN, SHORT, LOAD) are connected to the source port to be calibrated.
- [Defined THRU](#) or [Flush THRU](#) standard is connected between port pairs.
- Much quicker than SOLT when using a mechanical cal kit. ECal can also be used.

To select Enhanced Response:

For a standard S-parameter Cal, select **SmartCal** in the Cal Wizard.

Then, for all cals:

1. At the 'Select DUT Connectors page', check [Modify Cal](#), then click **Next**.
2. Under 'Cal Type', select **Enhanced Response**.

Enhanced Response cal also be selected as the default Cal Type using [Cal Preferences](#).

General Accuracy: High

Calibration Method: [SmartCal](#), [ECal](#)

Standards Required: (SHORT, OPEN, LOAD, [Defined THRU](#) or [Flush THRU](#))

Systematic Errors Corrected:

- Directivity (source port)
- Source match (source port)
- Isolation ([see exceptions](#))
- Load match (receiver port) - used only to produce transmission tracking term.
- Frequency response transmission tracking (receiver port).

- Frequency response reflection tracking (source port).
-

QSOLT (Quick SOLT)

Application: Used to quickly calibrate any number of ports. Developed specifically for use with [external multiport test sets](#).

Note: A [Delta Match Cal](#) is required to cal test ports that do not have a dedicated reference receiver.

- Reflection Standards (OPEN, SHORT, LOAD) are connected to only ONE of the ports to be calibrated. The lower port number of the ports to be calibrated is selected by default. This can be changed through the [Modify Cal / Cal Type](#) setting.
 - [Defined THRU](#) or [Flush THRU](#) standards are connected from the reflection standard port to the remaining ports to be calibrated.
 - Much quicker than SOLT when using a mechanical cal kit.
 - Based on TRL math.
-

General Accuracy: Not as high as SOLT

Calibration Method: [SmartCal](#), [ECal](#)

Standards Required: (SHORT, OPEN, LOAD, [Defined THRU](#) or [Flush THRU](#))

Systematic Errors Corrected:

- Directivity
 - Source match
 - Isolation ([see exceptions](#))
 - Load match
 - Frequency response transmission tracking
 - Frequency response reflection tracking
-

1-Port (Reflection)

Application: Used to accurately calibrate any single test port for reflection measurements only.

Calibration Method: [SmartCal](#), [Unguided Calibration](#), [ECal](#)

General Accuracy: High

Standards Required: (SHORT, OPEN, LOAD) or ECal module

Systematic Errors Corrected:

- Directivity
 - Source match
 - Frequency response reflection tracking
-

Open / Short Response

Application: Used to quickly calibrate any single test port for reflection measurements only.

Calibration Method: [Unguided Calibration](#)

General Accuracy: Low

Standards Required: OPEN or SHORT

Systematic Errors Corrected:

Frequency response reflection tracking

Thru / Transmission Response (Isolation Optional)

Application: Used to quickly calibrate any pair of test ports for transmission measurements only.

Isolation is not usually recommended. Learn more about [Isolation](#)

Calibration Method: [Unguided Calibration](#) and Guided Cal from the 'Select DUT Connectors page', check [Modify Cal](#), then click **Next**.

General Accuracy: Low

Standards Required: THRU

Isolation: One LOAD for each PNA test port.

Systematic Errors Corrected:

- Frequency response transmission tracking
 - Isolation
-

Last modified:

24-May-2013	Edit transmission resp
5-Sep-2008	Added note for ER Load Match
16-Apr-2008	Removed AR for TRL limitation
23-Feb-2007	Added Enhanced Response
12-Sept-2006	Added QSOLT

Calibration Wizard

The Calibration Wizard allows you to choose a Calibration method and then perform the calibration.

- [How to Start Calibration Wizard](#)
- [SmartCal \(Guided Calibration\)](#)
- [Unguided Calibration](#)
- [Saving a Calibration](#)

Other Cal Topics

How to start Calibration Wizard

Using front-panel HARDKEY [softkey] buttons

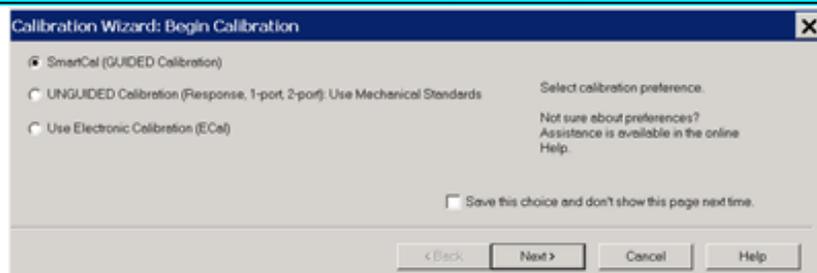
1. Press **CAL**
2. then [**Start Cal**]
3. then [**Cal Wizard**]

Using a mouse with PNA Menus

1. Click **Response**
2. then **Cal Wizard**

[Programming Commands](#)

Calibration Wizard Begin dialog box help



Select the calibration method:

[SmartCal \(Guided Calibration\)](#)

This method provides a step-by-step "wizard" interface. You describe the connectors on your DUT and the cal kits you will use; it walks you through the most accurate calibration possible.

- Supports ALL Cals **EXCEPT** simple open, short, and thru response Cals . See Also [TRL Calibration](#)
- Use a different Cal Kit (**including ECal**) for each port.

Unguided Calibration

This method provides a familiar calibration interface, but with limited capability. You choose the type of cal to perform; it allows you the flexibility to measure the standards in any order.

- Supports all Cals **EXCEPT** full 3-port, full 4-port.
- Only one Cal Kit can be used.
- Can NOT use Offset Load standard.

Why do we offer the Unguided Cal?

The Unguided Cal is familiar to legacy 8510 and 8753 customers. However, in those analyzers, you selected the cal kit, but not the connector type or gender. Therefore, the calibration engine did NOT know if the connectors were [insertable](#) or non-insertable. In the case of non-insertable, many people would use an undefined 'bullet' THRU and ignore the measurement inaccuracy.

The PNA Guided Cal Wizard asks for the DUT connectors and guides you through the most accurate calibration possible.

Use Electronic Calibration

- This method provides fast, software-controlled calibrations.
- Only one ECal module can be used. Use SmartCal when more than one ECal module is needed.
- See Also: [Perform a 4-Port Cal with a 2-Port ECal Module](#)

Save Preferences

- Clear to continue to see this page on subsequent calibrations.
- Check to save your calibration method choice and no longer see this page. To make this dialog re-appear, click **Response**, then **Cal**, then **Start Cal**, then **Cal Preferences**.
- Learn more about [Calibration Preferences](#).

The Calibration Window / Channel

During a Guided Calibration, a 'Cal Window' is created for you to view the connection of calibration standards before standards are measured. This Cal Window uses a new Cal channel that is created and duplicates the settings in the channel being calibrated. [Correction is ALWAYS OFF](#) for the displayed calibration channel. At the completion of the calibration, the calibration channel and window are deleted.

With PNA Rev. 7.50.27, the measurement of calibration standards can be performed while viewing any PNA window configuration you choose. The Cal Window is appended to your Custom Cal Window setting, and all windows are visible and sweeping below the Cal Wizard before the Measure (cal standard) button is pressed. The windows to be viewed and channels to be swept during the cal process are specified using [Remote commands](#). [See an example](#).

The new Cal Window settings do not work in a FCA channel.

SmartCal (Guided Calibration)

A Guided Calibration automatically determines the calibration type and suggests a calibration kit that matches your DUT connectors. Guided Calibration can perform the following Cal Types:

- ALL Cals **EXCEPT Open, Short, and Thru Response** Cals.
- ECal on one or more ports, beginning with [PNA firmware revision 5.24](#).
- TRL - [Learn how to do TRL cals](#)

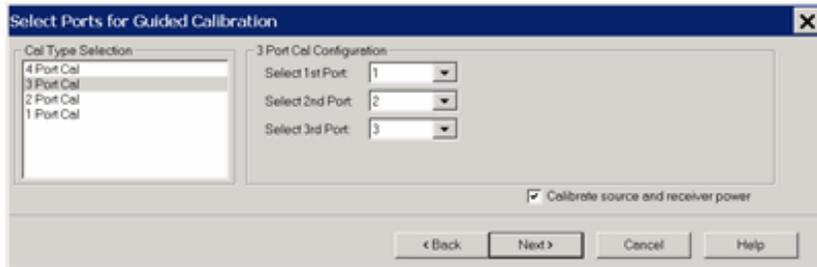
◀ Programming Commands ▶

Note: SmartCal DOES allow you to measure calibration standards in any order. However, you must click **Next** and **Back** without measuring standards until you get to the standard you want to measure.

The PNA displays the following dialog boxes when performing a Guided calibration on standard channels.

To learn about Calibrations for Application channels, refer to the help topic for the [Application](#).

Select Ports for Guided Calibration dialog box help



Allows you to select ports to calibrate.

Cal Type Selection Select the number of ports to calibrate.

N Port Cal Configuration If not calibrating all PNA ports, specify which ports to calibrate.

Calibrate source and receiver power Check to perform a Guided Power Calibration. [Learn more.](#)

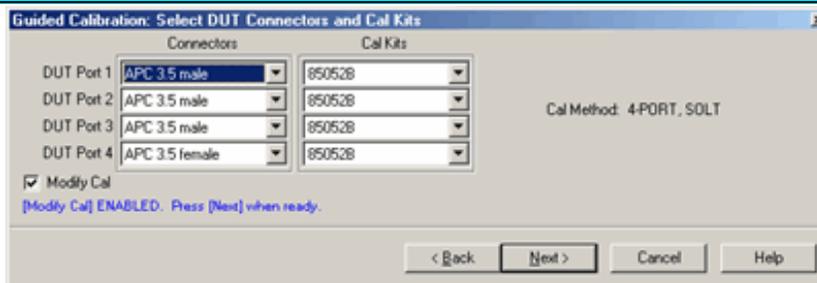
See Also: [Use Multiple Power Sensors to perform a Guided Power Cal](#)

Show Advanced Settings (Orientation & Thru Cal Section) Available only for [ECal](#).

Back Return to [Cal Wizard Begin](#) dialog. If you did not see the 'Cal Wizard Begin' dialog but want to, click **Back**, then clear the [Save Preferences](#) checkbox.

For greater than 4-port calcs, see [External Test Set calibration - Select Cal Type](#).

Select DUT Connectors and Cal Kits dialog box help



Allows you to select the connector type and Cal Kit for each DUT port to be calibrated.

Connectors To change selection, click the connector field for each DUT port.

If your DUT connectors are **not listed**, you can create your own connector type and calibration kit file. The PNA includes the following example cal kits that can be used as a template. See [Calibration kits](#) for more information.

- If using a gendered (male and female) connector type, select **Type A** as the connector type.
- If using a connectorless device such as on-wafer probes., select **Type B** as the connector type.

Cal Kits Select the Cal Kit to be used to calibrate each test port. The list for each DUT Port displays kits having the same connector type as the DUT.

Identical ECal models connected? ECal modules can be distinguished by serial number. This can have implications on your remote [SCPI](#) programs.

Cal Kit Notes

85056K

The 85056K definitions in the PNA are for 2.92mm standards (2.4mm plus 2.92 adapters). To calibrate 2.4 mm connectors using the 85056K cal kit, select 85056A as the cal kit when you need the sliding load. Otherwise, select 85056D as the cal kit. Both the 85056A and the 85056D kits contains exactly the same standards as the 85056K cal kit WITHOUT the adapters.

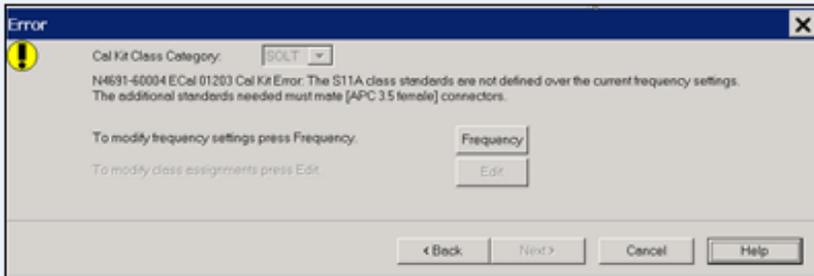
TRL

- To perform a [TRL Cal](#), assign a TRL Cal Kit to the lowest port number of each port pair.
- When selecting a TRL Cal Kit on a PNA-L model that requires a Delta Match Cal, and a [Global Delta Match Cal](#) is not available, the Cal type will be set to SOLT and a "Could not find a Global Delta Match Cal." message is displayed on the dialog box. If the selected Cal Kit will not support SOLT, the **Next** button will not be available. Then you must select a different Cal Kit to proceed or **Cancel** and perform a Global Delta Match Cal.

Modify Cal Check, then click Next, to [Modify Cal](#) (Standards AND Thru Method).

For greater than 4-port cals, see External Test Set calibration - Select DUT Connectors.

Error dialog box help



The current cal kit does not cover the current frequency range of the measurement. Do one of the following to correct the problem:

Cal Kit Class Category Choose from SOLT and TRL. Not available with ECal modules. Click **Edit** to modify the appropriate class assignments.

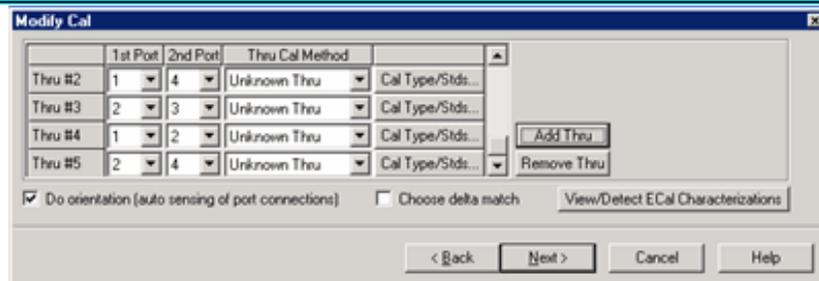
Frequency Change the frequency range of the active channel.

Edit Modify the class assignments so that a different standard is selected.

Back Select a different Cal Kit that covers the required frequency range.

Cancel Exit the Cal Wizard

Modify Cal dialog box help



Thru #n

Lists the proposed Thru connections to be made during the calibration process. You can change these Thru connections to better suit your test setup.

- The proposed Thru connections are listed automatically.
- Additional Thru connections can be selected for higher accuracy. [Learn more](#).

Add Thru

Click to add a Thru connection. [Learn more](#)

Remove Thru

Select a Thru by clicking the "Thru #N" field or the "1st Port / 2nd Port" field. Then click "Remove Thru". This selection is NOT available if the selected Thru is required for the calibration.

1st Port / 2nd Port

Click to select the two ports to be included in the Thru connection. The order of the port numbers is not critical.

Thru Cal Method

Lists the available Thru Cal methods for the specified port pairs.

[Learn about the Thru Cal Method choices.](#)

Cal Type/ Stds

Click to invoke the [View / Modify Properties of Cal dialog box](#)

Do orientation - Appears ONLY if an ECal module is selected for use.

When this box is checked (default) the PNA automatically senses the model and direction in which an ECal module port is connected to the PNA ports. If power to the ECal module is too low, it will appear as if there is no ECal module connected. If you use low power and are having this problem, clear this check box to provide the orientation manually.

Orientation occurs first at the middle of the frequency range that you are calibrating. If a signal is not detected, it tries again at the lowest frequency in the range.

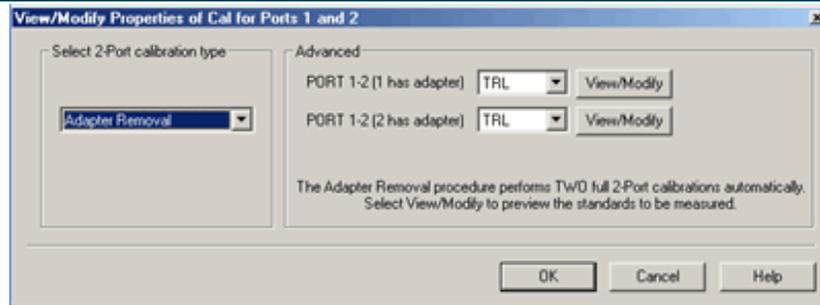
Choose delta match - Available when a Delta Match Cal is required.

- Check, then click **Next** to invoke the [Select Cal Set for Delta Match](#) dialog box.
- Clear - The Cal Wizard uses the [Global Delta Match Cal](#) if available.

View/Detect ECal Characterizations - Appears ONLY if an ECal module is selected for use.

Click to invoke the [View ECal Modules and Characterizations](#) dialog box. Displays a list of ECal modules that are connected to the PNA.

View/Modify Properties of Cal for Ports... dialog box help



Select calibration type

Another chance to change the Thru method.

[Learn about the Thru Cal Method choices.](#)

Advanced

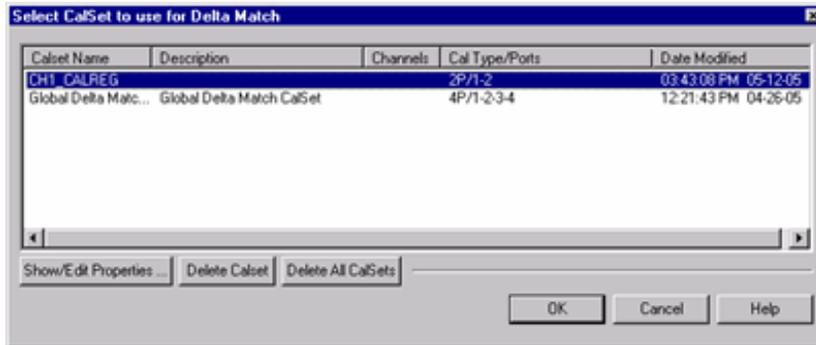
Select the cal method for each connector of the Thru pair.

- [TRL](#) - Available ONLY when a TRL cal kit was selected for the lowest port number of the port pair.
- [QSOLT](#) Available ONLY when "Defined Thru" or "Flush Thru" is selected. "**QSOLT 2 <= 1**" refers to the receive port 2 and source port 1 (where reflection standards are connected).
- [Enhanced Response](#) Available ONLY when "Defined Thru" or "Flush Thru" is selected. "**EnhResp 2 <= 1**" refers to the receive port 2 and source port 1.
- [Transmission Response](#) Available ONLY when "Defined Thru" or "Flush Thru" is selected, when Mechanical Cal is selected, and when 2 ports are being calibrated. "**TransResp 2 <= 1**" refers to the receive port 2 and source port 1.

View Modify Click to invoke the [Preview and Modify Calibration Selections](#) dialog box.

Note: Changes made to the Cal Kit through this dialog are **temporary** that last only for this calibration. To make permanent changes to the Cal Kit, perform [Advanced Modify Cal Kits](#).

Select Cal Set for Delta Match dialog box help



This dialog box appears when a Delta Match Cal is required and [Choose delta match](#) was selected. [Learn more.](#)

Displays the Cal Sets that meet the requirements of the Delta Match Cal.

Select either a User Cal Set or [Global Delta Match Cal](#).

If there is no suitable choice for a Delta Match Cal:

1. Click **Cancel**, then **Cancel** again to quit the Cal Wizard.
2. Perform either a [Global Delta Match Cal](#) or a SOLT cal and save the result in a User Cal Set.
3. Start the Cal Wizard to re-initiate this calibration.
4. Select the Global Delta Match Cal or User Cal Set.

Calibration Steps dialog box help



Note: Beginning in PNA Rev. 6.0, calibration can be performed with External triggers. [Learn more.](#)

As each new cal step prompt appears, the traces are setup for the next standard measurement. Also, sweeps are triggered continuously until the Measure button is pressed. This way you can view the integrity of the standard connection.

Prompts for standards to be measured.

Measure Click to measure the standard.

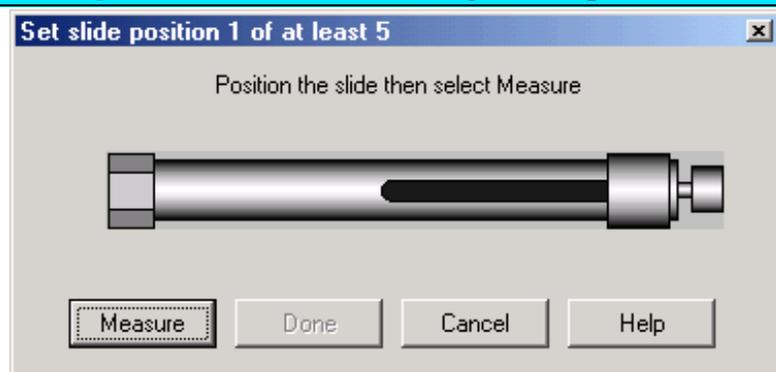
Done Click **after** a standard is re-measured and all measurements for the calibration are complete.

Next Click to continue to the next calibration step. Does **NOT** measure the standard.

If a standard is NOT measured, a warning appears and **Done** will not be available after the last Cal step.

Note: Smart (Guided) Cal allows you to measure calibration standards in any order. However, you must click **Next** and **Back** without measuring standards until you get to the standard you want to measure.

Sliding Load Measurement dialog box help



Allows you to measure the sliding load standard.

To ensure an accurate calibration, carefully follow the instructions that were provided with your sliding load.

To Measure a Sliding Load:

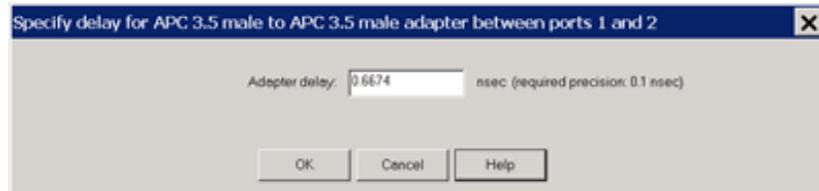
1. Connect the sliding load to the measurement port.
2. Position the sliding element, then click **Measure**. Do not move the sliding element until measurement is complete.
3. Measure the sliding load for at least **five** and up to **seven** positions for best accuracy.

Note: The positions of the sliding element should cover the full length of the slide, but be unequally spaced to reduce the possibility of overlapping data points. Most sliding loads have

marks for each slide position.

4. Click **Done** after the final measurement.
5. Remove sliding load from the measurement port.
6. Measure the remaining standards.

Specify delay dialog box help



This dialog appears ONLY when [Adapter Removal](#) or [Unknown Thru](#) calibrations are performed.

The following values were estimated from the measurement. Most of the time, they are adequate. However, for CW sweep or frequency sweep with large step sizes, the accuracy of the values may be improved.

Adapter delay To improve this value, measure and record the delay of the adapter with a dense step size. Enter that value here. The required precision value is the accuracy that is required to characterize the delay value.

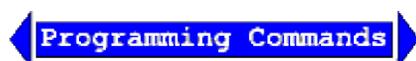
Nominal phase offset (Waveguide ONLY). To improve this value, measure and record the phase offset of the Waveguide adapter with dense step size. Enter that value here.

When one connector is coax and the other connector is waveguide, the phase offset has an ambiguity of 180 degrees. For consistency, the estimate provided here is always between 0 and 180 degrees. You can change this estimate to any value between -180 degrees and +180 degrees.

For FCA/Mixer calibrations, this dialog box appears twice: once for the input frequencies and once for the output frequencies. The values can be slightly different.

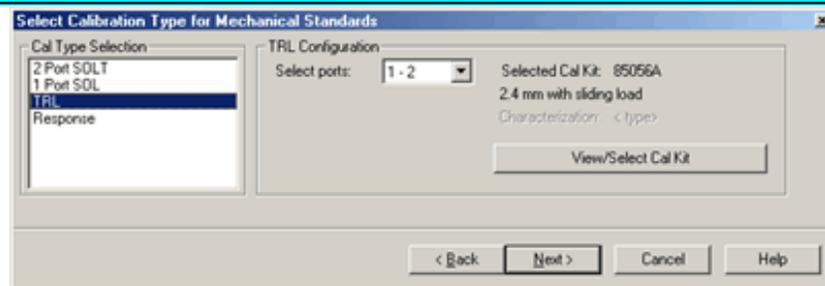
The [Calibration Complete](#) dialog box appears after all standards are measured.

Unguided Calibration



The PNA displays the following dialog boxes when performing an Unguided calibration:

Select Calibration Type for Mechanical Standards dialog box help



Unguided calibration does **NOT** support cals greater than 2 ports or **E**Cal calibrations.

Calibration Type Selection

- 2-Port SOLT
- 1-Port SOL
- TRL - [Learn more about TRL](#)
- **Response** - Reflection and Thru (if the active measurement is transmission)

Cal Configuration If not calibrating all PNA ports, specify which ports to calibrate.

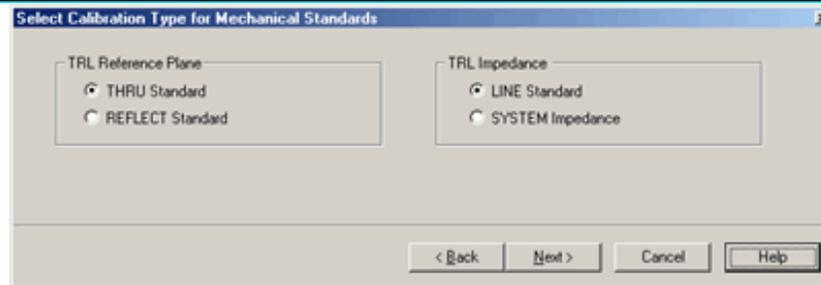
Back Return to [Cal Wizard Begin](#) dialog. If checked, you can clear the [Save Preferences](#) checkbox to see the Begin page when the Cal Wizard begins.

View/Select Cal Kit Click to invoke the [Select Cal Kit dialog box](#).

Note: Beginning with A.09.50, when selecting a Cal Kit with an impedance other than 50 ohms (Waveguide = 1 ohm), it is **NO LONGER NECESSARY** to change the [System Impedance](#) setting before performing a calibration. The impedance for the calibration is now derived from the Cal Kit impedance.

Next Click to continue to [Measure Mechanical Standards](#) dialog box.

Select Cal Type dialog box help



This dialog box only appears if the selected Cal Type is TRL in the previous dialog box.

TRL Reference Plane Select which standard to use to establish the position of the measurement reference plane.

THRU Standard Select if the THRU standard is zero-length or very short.

REFLECT Standard Select if the THRU standard is not appropriate AND the delay of the REFLECT standard is well defined.

TRL Impedance

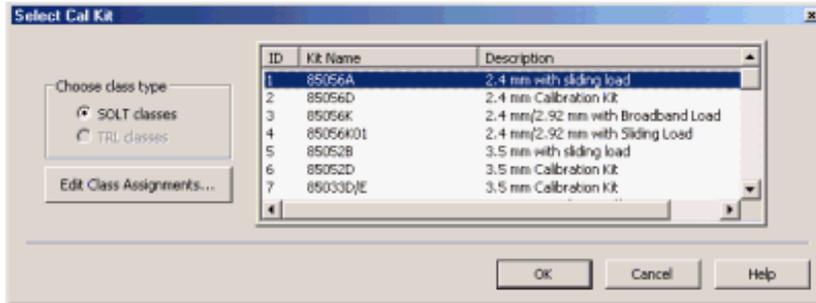
LINE Standard Specifies that the characteristic impedance of the LINE standard should be used as the system impedance. This ignores any difference between Offset Z0, Offset Loss, and System Z0.

SYSTEM Impedance Transforms the LINE standard impedance and loss to that of the system impedance for use with the calibration error terms. The TRL calibration will first compute the error terms assuming the LINE standard impedance is the system's characteristic impedance (same as previous LINE selection), then modify the error terms to include the impedance transformation. This should only be used with coax since the skin effect model used is a coaxial model.

[Learn how to change System Z0.](#)

To learn to substitute other calibration kits, see [Advanced Modify Cal Kits](#)

Select Cal Kit dialog box help



Displays the calibration kit files available for Unguided calibration. Select the desired calibration kit file and click **OK**.

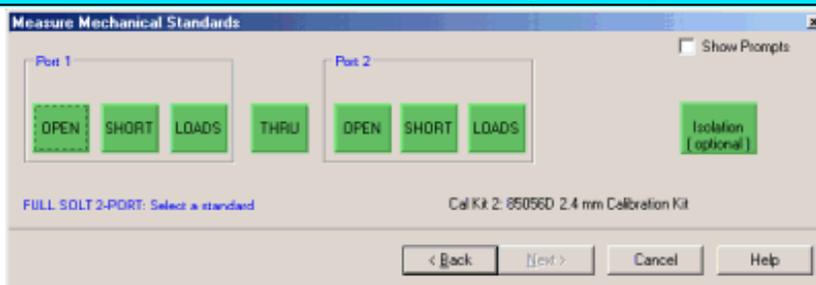
Choose class type

Edit Class Assignments Allows modification of the selected Cal Kit class assignments.

- To learn to substitute other calibration kits, see [Advanced Modify Cal Kits](#)
- Unguided Cal can access only mechanical cal kits #1 through #95, although more cal kits can imported. [Learn how.](#)

Note: Beginning with A.09.50, when selecting a Cal Kit with an impedance other than 50 ohms (Waveguide = 1 ohm), it is **NO LONGER NECESSARY** to change the [System Impedance](#) setting before performing a calibration. The impedance for the calibration is now derived from the Cal Kit impedance.

Measure Mechanical Standards dialog box help



Note: Beginning in PNA Rev. 6.0, calibration can be performed with External triggers. [Learn more.](#)

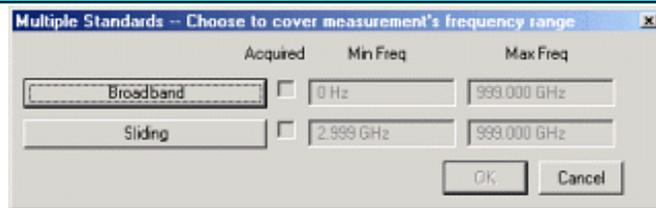
Displays the calibration kit file and standards required for the calibration.

- Standards may be connected and measured in any order.

- Connect the standard to the measurement port and click its associated green button. A check mark indicates the standard has been measured.
- If a standard type contains multiple standards, the [Multiple Standards dialog box](#) opens to display the multiple standards included in the calibration kit file.
- If a sliding load is included in the calibration kit file, the [Sliding Load dialog box](#) opens to perform the measurement with the standard.
- **Reflection Response** Select EITHER Open or Short standard, then click **Next**.
- **Isolation** Requires one load for each test port of the PNA. [Learn more about Isolation](#). Use when your measurement requires maximum dynamic range (> 90 dB). See also [Isolation Portion of 2-Port Calibration](#).
- **Normalize** Available when performing a response cal for any measurement. After Normalize is pressed and the Cal is complete, the data trace is flat when the same physical connections are present on the port. This is similar to [Data/Memory](#), except that the response cal is [saved with Cal data](#) and can be applied to other like measurements. Data/Memory is still available after using Normalize. You would usually connect a THRU standard when calibrating a transmission measurement, and a SHORT standard when calibrating a reflection measurement.

Show Prompts Check to provide a reminder for the required connection when you click on the standard.

Multiple Standards dialog box help



Select the standards to be measured.

Note: You may see both male and female standards. The Unguided cal has no knowledge of the gender of your connector types. **Choose the gender of your DUT connector; NOT the test port.** Then click OK.

To modify this calibration class to show only one standard, on the Calibration menu, click **Advanced Modify Cal Kits**. Select the Cal kit and click **Edit Kit**. In **Class Assignment**, click **Edit**. Learn more about [Modify Calibration Class Assignments](#).

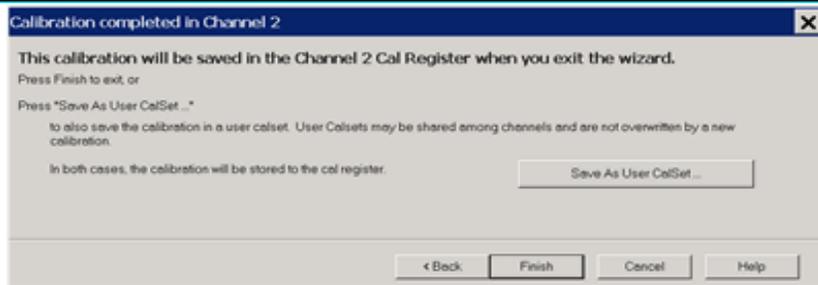
- Connect the standard to the measurement port and click its associated button. A check mark in the **Acquired** box indicates the standard has been measured.
- To cover the entire frequency range, you may need to measure more than one standard. The order in

which the standards are measured is important. The last standard that is measured will override the others in respect to the frequency range of the standard definition. **Example:** In the case of measuring both a broadband load and a sliding load, you would measure the sliding load last. This is because the frequency range of the sliding load is a subset of the broadband load.

Saving a Calibration

SmartCal, ECal, and Unguided Calibrations end with the following dialog box:

Calibration Completed dialog box help



Finish Save to the channel's calibration register.

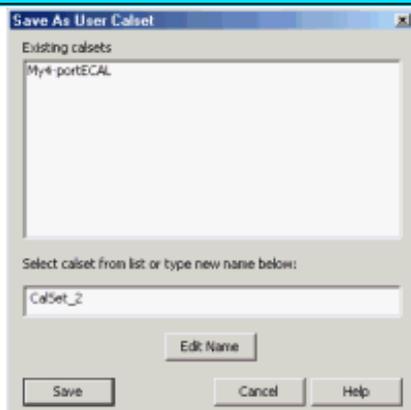
Save As User Cal Set Invokes the [Save as User Cal Set dialog box](#) AND save to the channel's calibration register.

Cancel Calibration is NOT applied or saved.

Learn about [Calibration Registers](#).

Learn about [User Cal Sets](#)

Save as User Cal Set dialog box help



Existing Cal Sets - Lists the Cal Set names saved on the PNA.

Select Cal Set from list or type new name below Specify a name for the new Cal Set. Either accept the suggested new name, type a new name, or select a name from the list to overwrite an existing name.

Edit Name If there is no keyboard, click to start the PNA typing tool that can be used from the PNA front panel.

Save Saves the Cal Set to the new Cal Set name.

Learn about [User Cal Sets](#)

Last modified:

11-Jul-2013	Removed link to sliding load in Cal Kit
24-May-2013	Added TransResp to guided selection
14-Nov-2012	Added Unguided explanation
25-Oct-2012	Edited for DMC requirements - removed limitation of TRL in Unguided Cal
12-Apr-2012	Unguided impedance note
12-Oct-2011	Edit Cal kit limitation
14-Jul-2011	Removed WG impedance restriction for Guided
16-Sep-2010	Added note for TRL and Noise Cal
3-Sep-2008	Removed legacy content
14-Apr-2008	Added note about Offset Load
4-Mar-2008	Added Cal Window feature
21-Sep-2007	Added note about no TRL on 4-port PNAs
20-Jan-2007	Added note about any order for SmartCal.
18-Sept-2006	MQ Major modifications for multiport

Guided Power Calibration

Beginning with PNA firmware release A.09.30, Source and Receiver Power Calibration can be performed during a standard S-parameter Guided Calibration. This power cal provides the following enhancements over the standard source and receiver power calibration:

- A source and receiver power cal can be performed for all PNA ports with a single power sensor connection.
- [Multiple power sensors](#) can be used to cover wide frequency ranges.
- The receivers are corrected automatically.
- Optionally compensates for an adapter that may be used to connect the power sensor.
- Provides [optional match-corrected power measurements](#).
- Source and Receiver power correction is stored to the Cal Set along with S-parameter correction.

Note: A Guided Power Calibration is not accurate when [Frequency Offset Mode](#) is enabled.

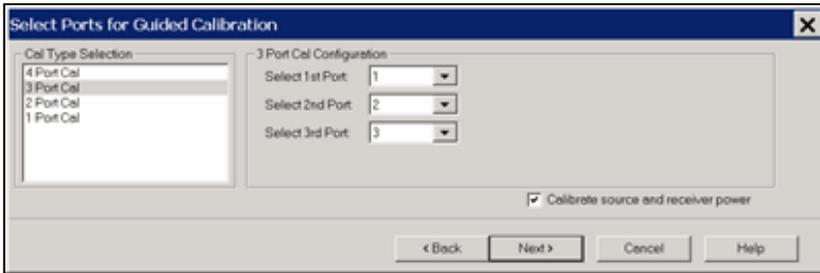
Learn more about the standard [Source](#) and [Receiver](#) Power Cals.

How to perform a Guided Power Cal

1. In a [Standard \(S-parameter\)](#) channel, setup your measurements (sweep type, frequency range, IFBW, and so forth). A special version of this feature is available on mmWave SMC measurements. [Learn more](#).
2. Connect the Power Meter / Sensor the same as a standard Source Power Cal. [Learn more](#).
 - See [Supported Power Meters](#)
 - See [Important first-time USB connection note](#).
3. Start the **Cal Wizard**, then select **Guided (Smart) Cal**. [Learn how](#).

◀ Programming Commands ▶

On the following **Select Ports** dialog, check **Calibrate source and receiver power**, then click **Next**.



Two Cal Wizard pages later, complete the following dialog.

Power Cal Settings dialog box help

Note: A **Use Power Table** checkbox (not shown) is available when a mmWave SMC measurement is active. [Learn more](#).

Power Cal at: Select the source port for which a Power Calibration will be performed. The source and receiver correction will be transferred to all other sources and receivers involved in the S-parameter measurements.

Use Multiple Sensors NOT available with SMC measurements.

Check this box to use one or more power sensors that are [configured as PMAR devices](#). This dialog is replaced with the [Multiple Sensors](#) dialog. See following image.

When "Use Multiple Sensors" is cleared (default setting), click [Power Meter Settings](#) to configure the power meter.

De-embed (power sensor) adapter When the power sensor connector is NOT the same type and gender as the DUT connector for the specified port, then for optimum accuracy, extra cal steps are required to measure and correct for the adapter that is used to connect the power sensor to the reference plane.

Clear this box to NOT compensate for the added adapter.

Check this box to perform extra calibration steps to measure and correct for the adapter.

Then select the **Power Sensor Connector** type and gender of the power sensor. "Ignored" does NOT compensate for the added adapter, just as if the checkbox were cleared.

When this connector matches the DUT connector for the same port, then the PNA assumes that there is no adapter. Extra cal steps are NOT required and the Cal Kit selection is not available.

Otherwise, select the **Cal Kit** to be used to calibrate at the adapter.

See [Accuracy Settings](#) below.

Power Cal Settings - Use Multiple Sensors dialog box help

Sensor	Start	Stop	Adapter	Connector	Cal Kit	
NB487A	10.000000 MHz	50.000000 GHz	<input checked="" type="checkbox"/> Use	1.00 mm female	85058A Databased	Remove
VB486A	50.000000 GHz	75.000000 GHz	<input checked="" type="checkbox"/> Use	1.00 mm female	85058A Databased	Remove
WB486A	75.000000 GHz	110.000000 GHz	<input checked="" type="checkbox"/> Use	1.00 mm female	85058A Databased	Remove

Notes

"Multiple sensors" are allowed ONLY on standard channels and during a [Cal All calibration](#).

The power sensors that are used as "multiple sensors" MUST be configured [PMAR devices](#).

Power Cal at: Select the source port for which a Power Calibration will be performed. The source and receiver correction will be transferred to all other sources and receivers involved in the S-parameter measurements.

Sensor Settings Click to start the Sensor Settings dialog, used to [ADD / Configure an External Device](#).

Sensor Grid

Sensor Select the power sensor and the associated **Start** and **Stop** frequency range.

Adapter When the power sensor connector is NOT the same type and gender as the DUT connector for the specified port, then for optimum accuracy, extra cal steps are required to measure and correct for the adapter that is used to connect the power sensor to the reference plane.

Clear this box to NOT compensate for the added adapter.

Check this box to perform extra calibration steps to measure and correct for the adapter. Then

specify the **Power Sensor Connector** type and gender of the power sensor. When this connector matches the DUT connector for the same port, then extra cal steps are NOT required, and the Cal Kit selection is not available. Otherwise, select the **Cal Kit** to be used to calibrate at the adapter.

Remove Click to remove the power sensor from the list.

Add Sensor Click to add a new line, then click the down-arrow to select a sensor. If a power sensor does NOT appear in the list, click the **Sensor Settings** button to configure a power sensor.

Accuracy

Tolerance When consecutive power sensor readings are within this value of each other, then the reading is considered settled.

Max Number of Readings Sets the maximum number of readings the power sensor will take to achieve settling. Each power reading is "settled" when either:

- Two consecutive readings are within this **Tolerance** value or
- When the **Max Number of Readings** has been met.

The readings that were taken are averaged together to become the "settled" reading.

Programming Commands

Power Sensor Connection step dialog box help



Power Level Set the power level at which the Source Power Cal is to be performed.

It is usually best to perform the Source Power Cal at 0 dBm because the power sensor is calibrated at that level. If 0 dBm is not achievable for your measurement, then set to the power level with the lowest level of measurement noise.

Turn OFF Match Correction

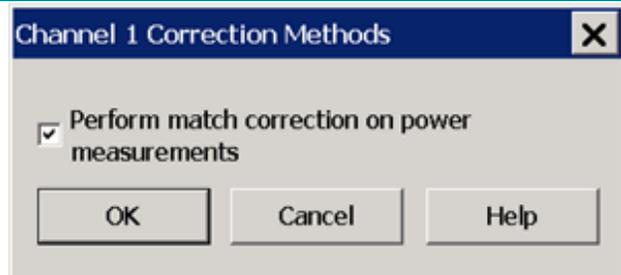
During a Guided Power Cal, the match between the power sensor and the PNA source port is

measured. The source power correction array is compensated to account for the measured mismatch. In addition, the reference receiver measurement is also compensated to account for the mismatch of the DUT.

How to turn OFF match correction:

Click **Response**, then **Cal**, then **Manage Cals**, then **Correction Methods**.

Correction Methods dialog box help



By default, the Guided Power Cal applies match correction to the receiver measurements. However, you may want to turn OFF match correction in the following cases:

- When making non-traditional measurements, such as high-power or multiport configurations. Because of added components or reconfigurations, the mismatch measurement may not be valid.
- When you have a remote program that already accounts for the match effects of the sensor.

Clear the box to NOT apply match correction on all receivers used on this channel. The receivers are still corrected using standard receiver calibration. [Learn more](#).

Check the box to apply match correction on all receivers used on this channel.

Last Modified:

4-Apr-2013	Added Cal Set save note
1-May-2012	Edit for Cal All
5-Apr-2012	Minor edits
12-May-2011	Modified for mmWave SMC
21-Dec-2010	Added multiple power sensors
31-Aug-2010	MX New topic

Calibrate All Channels

"Cal All" allows you to calibrate multiple channels in a single calibration session. This not only reduces the number of connections that need to be made, but also the number of cal standard measurements that must be performed.

In this topic:

- [Features](#)
- [Limitations](#)
- [How to perform a Cal All Channels Calibration](#)
 - [Select Channels dialog](#)
 - [Measurement Class Cal Properties dialog](#)
 - [Calibration Attenuator Settings dialog](#)
 - [Select DUT Connectors and Cal Kits dialog](#)
 - [Power Cal Settings dialog](#)
 - [Cal Steps dialog](#)
 - [Finish](#)

Other Cal Topics

Features

Cal All offers a single, optimized calibration procedure for all channels (with some limitations, see below). The optimizations include:

- Minimizing the number of physical connection of standards.
- Minimizing the number of power meter calibration sweeps.
- User-settable power levels for S-Parameter as well as power calibration steps.
- Accounting for different switch and attenuator settings among different channels. This reduces the number of measurements required to characterize different switch/attenuator settings (channel setup differences).
- Cal All will produce the same number and format of Cal Sets (error terms) that would be realized had the calibrations been performed one at a time.

Limitations

- [VMC](#) channels are NOT supported.
- [SMC+ Phase](#) with phase enabled is supported using a known delay mixer or a phase reference cal set. S2P file characterized mixers are NOT supported.
- For non-IMD channels, the wideband IF path is used during Cal All. Therefore, non-IMD channels that use the narrowband IF path (for [Narrow band pulse](#) measurements for example) or have [manually-selected IF frequencies](#) will not be properly calibrated.
- [ECal Extrapolation](#) is NOT supported on Swept IMD and IMDx channels.
- Starting the PNA in [Multiport mode](#) is NOT supported.
- Cal All is performed at one IFBW.
- All channels that are calibrated are forced into [stepped sweep mode](#).
- All channels to be calibrated MUST have the same [cal reference plane](#). In other words, Cal All cannot compensate for any path changes that occur external to the PNA.

How to perform a Cal All Channels Calibration

Using front-panel HARDKEY [softkey] buttons

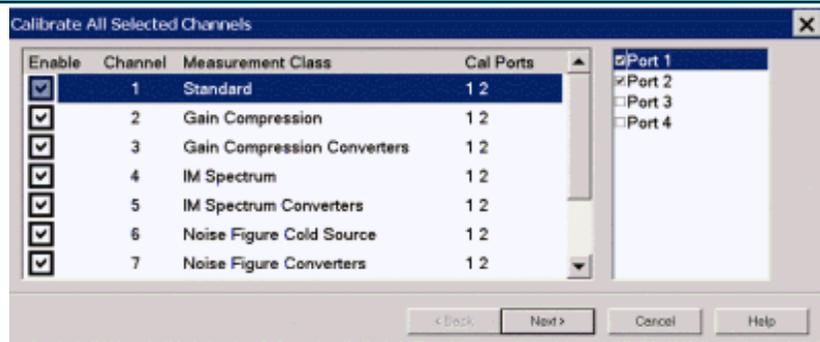
1. Press **CAL**
2. then **[Start Cal]**
3. then **[Cal All Channels]**

Using a mouse with PNA Menus

1. Click **Response**
2. then **Cal**
3. then **Start Cal**
4. then **Cal All Channels**

Programming Commands

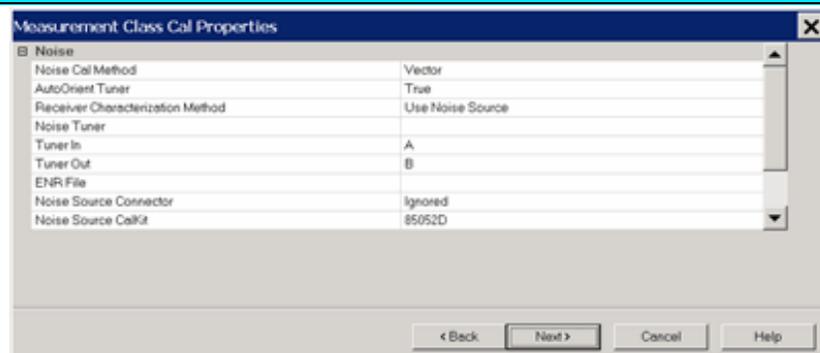
Selected Channels dialog box help



1. Check the channels to be calibrated.
2. Check the ports to be calibrated.
3. Click **Next>**

Note: To perform an LO power cal for a mixer channel, set the LO port to a PNA or external source in the [Mixer Setup dialog](#). Then select that port in this dialog.

Measurement Class Cal Properties dialog box help



Confirm or change the following unique cal properties for each channel to be calibrated. Click a link to learn about these properties.

The properties with **(NOT available in Cal All)** are NOT available in a Cal All calibration as they are in a stand-alone calibration.

Gain Compression and GCX

- No unique properties.

Noise Figure and NFX

	Programming	
UI Setting	Property	Value
Cal Method	"Noise Cal Method"	"Scalar" "Vector"
Noise Tuner (used with Vector Cal Method)	"AutoOrient Tuner"	"true" - AutoOrient "false" - Do NOT autoOrient
When "AutoOrient Tuner" = false (Manually Orient)	"Tuner In" "Tuner Out"	Tuner port connected to DUT input Tuner port connected to DUT output For example: "A" or "B"
Receiver Characterization Method	"Receiver Characterization Method"	"Use Power Meter" "Use Noise Source"
Specify the Noise Source connector - used for the 1 port cal if needed.	"Noise Source Connector"	String name of valid connector type. To query valid connector types: COM: ValidConnectorTypes SCPI: SENS:CORR:COLL:GUID:CONN:CAT?
Specify the Noise Source cal kit - used for the 1 port cal if needed.	"Noise Source CalKit"	String name of valid cal kit. To query valid cal kits names: COM: GetCompatibleCalKits SCPI: SENS:CORR:COLL:GUID:CKIT:CAT?
Specify ENR file	"ENR File"	String path and filename of the ENR file.
Include Power Calibration	"Include Power Calibration"	"true" or "false"

Swept IMD and IMDX

	Programming	
UI Setting	Property	Value
Max Product Order	"Max Product Order"	Integer indicating desired product order.
Include 2nd Order	"Include 2nd Order"	"true" or "false"
Response only (NOT available in Cal All)		
Center Frequencies only (NOT available in Cal All)		

IM Spectrum

- No unique properties.

SMC (VMC is NOT offered)

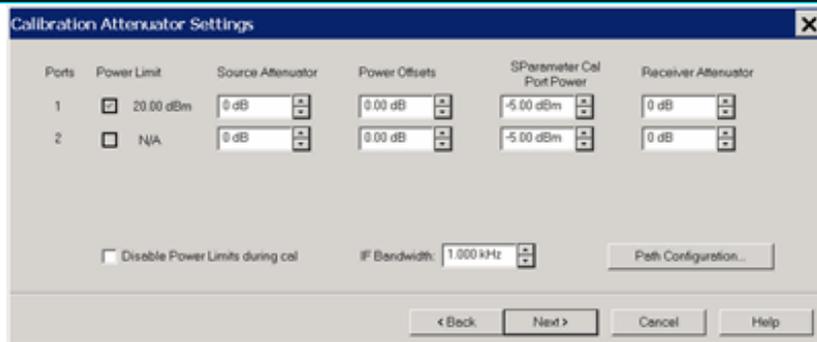
	Programming	
UI Setting	Property	Value
Enable Phase Correction	"Enable Phase Correction"	"true" or "false"
Phase Correction Method	"Phase Correction Method"	"Use Mixer Delay" "Use Receiver Characterization Calset"
Mixer Delay	"Mixer Delay"	Real number indicating delay value.
Receiver Characterization Cal Set	"Calset"	String of Cal Set Name
Characterized mixer (s2p file) (NOT available in Cal All)		

Standard Channel

	Programming	
UI Setting	Property	Value
Include Power Calibration	"Include Power Calibration"	"true" or "false"

The power cal is optional only if none of the selected channels require a power cal.

Calibration Attenuator Settings dialog box help



This dialog shows the Power, Attenuator, and IFBW settings for the Cal All calibration. The default values for the Cal All session are the preset values of a standard S-parameter channel. These values are not necessarily the same as those of the channels that are selected for calibration. When there are differences in measurement path (switch) settings between the Cal All channel and the selected channels, these differences are detected by Cal All and additional measurements are made for each path condition. These additional measurements allow Cal All to produce error terms appropriate for each of the selected channels. In general, the Cal All session should be performed at a power level that is high enough to prevent noise in the error terms. However, an increase in power could cause compression or damage to the PNA receivers. The following settings allow you to increase the power level ONLY during the Cal All session.

Power Limit (Disable)

Cal All shows you when power limits are enabled. This setting provides you a convenient way to TEMPORARILY disable these limits in order to take advantage of the power settings available in Cal All. If power limits are on, your DUT is probably a high-gain device and the attenuator settings in your channels are high resulting in lower power at the cal reference plane. This lower signal can result in noisier measurements during the acquisition of cal. This situation is precisely what Cal All is intended to improve. Cal All allows you to configure the calibration conditions for better signal-to-noise performance during the cal while leaving your DUT conditions alone. You can elect to clear the “Disable Power Limits during cal” checkbox when you prefer to calibrate at a higher power level than is allowed by your limit. The limit is restored after the Cal All session.

Source / Receiver Attenuator

By default, the Cal All calibration is performed with Source and Receiver attenuators set to 0. Change the Source or Receiver attenuator settings when external hardware (such as a booster amplifier) would cause the PNA receivers to be compressed or damaged.

You may also want to change the attenuator or path configuration settings to force the cal channel

to match settings of the selected channels. If all of the selected channels are set to identical hardware settings, it may be better to apply these settings to the cal channel. For example, if your channels all use a 5 or 10 dB attenuator step at port 1, you might elect to change the Cal All channel to use the same low attenuator settings. This will result in the cal measurements being made under the same path conditions as the channel and it will eliminate the need to mathematically compensate for the difference. However, if large attenuator values are used, the default Cal All settings will likely improve your results.

S-Parameter Cal Port Power

Set the power level at which the S-Parameter cal is performed.

Power Offsets

Power Offsets are channel-scoped. Consequently, offsets that you already set are NOT automatically copied to the Cal All session. This setting allows you to also apply a Power Offset during the Cal All session. [Learn about Power Offsets.](#)

IF Bandwidth

Set the IFBW used to perform the Cal All calibration. The default IFBW setting of 1 kHz is a good nominal setting for most measurements. Lowering the IFBW removes noise from the calibration measurement, but also causes slower sweeps.

Path Configuration

In addition to application-specific calibration requirements, Cal All performs an S-parameter calibration + power cal on behalf of all the selected channels. By default, this calibration is performed with standard channel default hardware paths:

- Bypass switches are set to “Thru”
- Noise switches are set to Normal.

Path configuration differences between your channel and the Cal All channel are mathematically removed.

The [Path Configuration dialog](#) allows you to change the path settings for this portion of Cal All calibration.

Select DUT Connectors and Cal Kits dialog box help



For each DUT port:

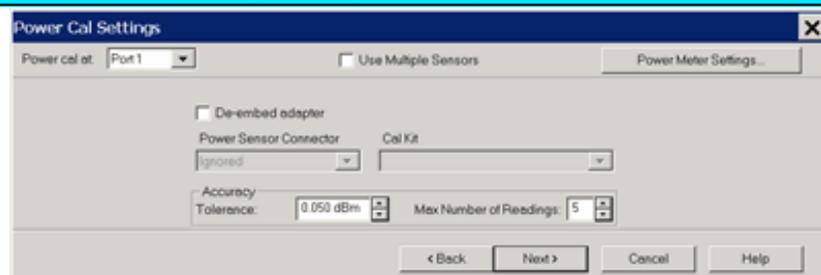
- Select the connector at the calibration reference plane (where the cal standards will be connected).
- Select the cal kit to be used.

Check **Modify Cal** to change the Thru method. An Unknown Thru cal is performed by default.

[Learn about THRU methods.](#)

[Learn more about this dialog.](#)

Power Cal Settings dialog box help



A guided power cal is performed on the source ports for the Cal All calibration.

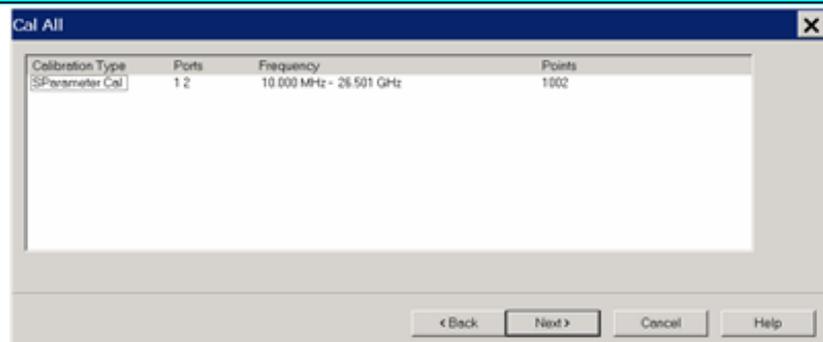
This dialog is displayed for each source port to receive a power cal.

To perform an LO power cal for a mixer channel, set the LO port to a PNA or external source in the [Mixer Setup dialog](#). Then select that port in the [Selected Channels dialog](#).

- To use the **same** power sensor for all power cals, do **NOT** check Use Multiple Sensors.
- To use **different** power sensors, check **Use Multiple Sensors**. The sensor must be configured as a PMAR device. [Learn how.](#)

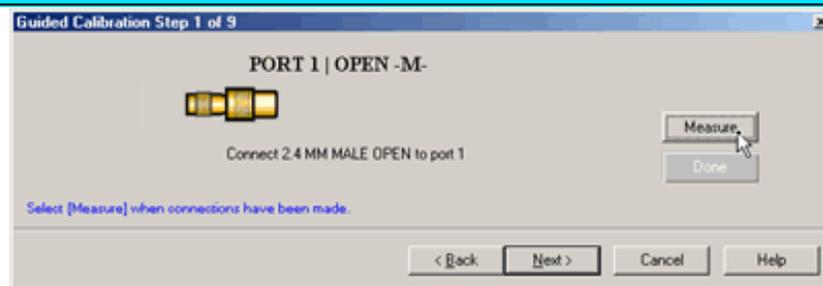
[Learn about this dialog box.](#)

Cal All Summary dialog box help



This page is a summary of the Cal All settings. Confirm the settings, then click **Next >** or **< Back** to change settings.

Cal Steps dialog box help

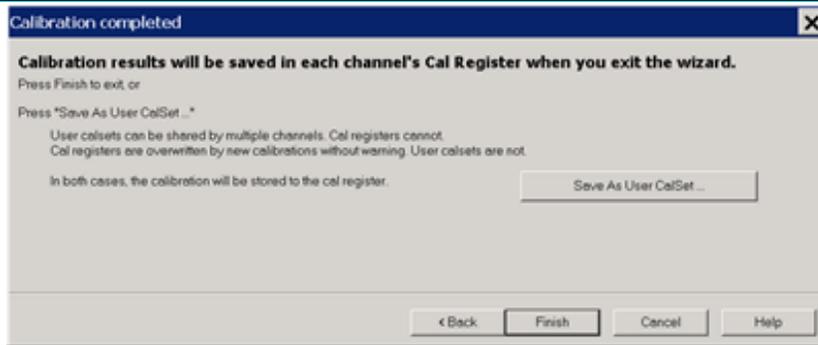


Follow the prompts to connect each standard. Then click **Measure**.

Click **Re-measure** if necessary.

Then click **Next >**

Finish Cal dialog box help



Click **Finish** to save the Cal All session results to Cal Registers.

Or click **Save As User CalSet**, then enter a prefix title. The Meas Class and channel number are appended to this prefix to save to a User Cal Set for each calibrated channel.

[Learn more about this dialog.](#)

Last modified:

10-Feb-2014 Added ECal Extrap limitation
10-Feb-2014 New topic

Using Calibration Sets

- [What are PNA Cal Sets](#)
- [Cal Registers and User Cal Sets](#)
- [How to Manage and Apply Cal Sets](#)
- [Examples of Cal Set Usage](#)
- [Archiving Cal Sets using .cal files](#)

See Also

Save and Recall: [Instrument States and Cal Set Data](#)

[See other Calibration Topics](#)

What are PNA Cal Sets

At the completion of a calibration, all calibration data is stored to a Cal Set. The Cal Set can be applied later to any channel that has the same stimulus settings as the Cal Set, thereby saving the time it takes to perform another calibration. The following data is saved to a Cal Set:

- Name
- Cal Set Description
- Cal Set Attributes - stimulus settings, cal type, port association
- Standards data
- Error term data
- GUID (**G**lobally **U**nique **I**dentifier)

Cal Registers and User Cal Sets

There are two types of Cal Sets:

- **Cal Registers** (channel specific)
- **User Cal Sets**

Calibration data is automatically saved to a Cal Register at the end of every calibration. You can also

choose to save the cal data to a User Cal Set.

Calibration Registers

New with PNA Release 5.0, Calibration Registers are designed to simplify calibrations for most users. When a calibration is complete, the data is automatically saved to the channel's Cal Register, overwriting (or [appended to](#)) the previous cal data stored in that register. This concept is similar to previous Agilent Vector Network Analyzers.

- Every channel has ONE dedicated Cal Register. They are named CHn_CALREG, where n is the channel number. The name cannot be changed.
- Cal Registers are more volatile because they are overwritten (or [appended](#)) each time a calibration is performed on that channel. The Cal data is always saved, but only temporarily.
- Cal Registers can be applied to other measurements, but ONLY on the same channel as the Cal Register.

User Cal Sets

At the end of a calibration, you can choose to also save cal data to an existing or new User Cal Set.

- User Cal Sets can be applied to any number of channels simultaneously.
- User Cal Sets are named by you for easy identification.
- You can have an unlimited number of User Cal Sets.
- At any time, you can copy Cal Register data to create a User Cal Set. See [Cal Set Properties](#).

Note: You can [set a Preference](#) to revert the pre-5.0 behavior - Cal data was automatically saved to a User Cal Set instead of a Cal Register.

Appending Data in a Cal Set

At the end of a calibration, data is saved to the channel's Cal Register and, if you choose, to a User Cal Set. When you choose to save to an existing User Cal Set, the PNA attempts to append the new error terms to the existing User Cal Set. The existing Cal Set data is completely overwritten UNLESS the new data can coexist with the existing data according to the following two rules:

- The stimulus settings of the new data must exactly match the existing data.
- The new cal must involve different ports from the existing cal.

For example:

Case 1 - An existing Cal Set contains a full 2-port cal between ports 1 and 2. Using the same stimulus settings, you perform a 1-port cal on port 3. At the end of the cal, you click [Save As User Cal Set](#) and

select the existing full 2-port User Cal Set.

Result: The 1-port cal is appended to the 2-port User Cal Set. There is NO overlap between them.

Case 2 - Same situation as Case 1, except the 1-port cal is performed on port 1.

Result: The Cal Set will contain a 1 port cal on port1 and a 1 port cal on port 2. The overlapping tracking terms are removed rendering the original full 2 port cal invalid.

How to Manage and Apply Cal Sets and Cal Types

The PNA attempts to apply a Cal Set, and turn error correction ON, for ALL of the measurements on the active channel. This may not always be possible. For example, suppose a channel contains both S11 (reflection) and S21 (transmission) measurements. If a Cal Set that contains only an S11 **Cal Type** is applied to that channel, the Cal Set does not contain the error terms to correct the S21 measurement. Error correction is turned ON for the S11 measurement and NOT turned on for the S21 measurement.

There are two ways to apply an existing Cal Set (Cal Register or User Cal Set) to a measurement:

1. Recalling an Instrument State with Cal data ([.cst file](#)) - A .cst file contains an Instrument State with all measurement attributes AND a 'pointer' to the Cal Set that was used to calibrate the measurement. Before saving a .cst file, be sure that a User Cal Set (NOT a Cal Register) is being used for the measurement. Because Cal Registers are automatically overwritten when a new calibration is performed, it is likely that the Cal Register data will change before the .cst file is recalled.
2. Create a new measurement and select a Cal Set to apply to the active channel.

Note: NEVER copy or modify Cal Sets from Windows Explorer or other applications. Cal Sets should only be accessed through the PNA Application.

How to select and apply a Cal Set to the active channel

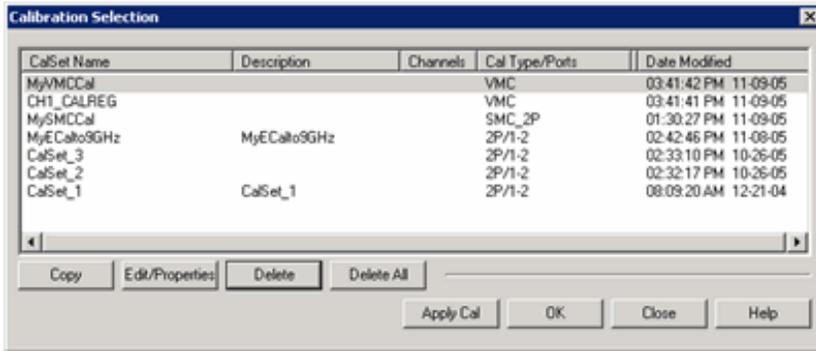
Using front-panel HARDKEY [softkey] buttons

1. Press CAL
2. then **[Manage Cals]**
3. then **[Cal Set]**

Using a mouse with PNA Menus

1. Click **Response**
2. then **Cal**
3. then **Manage Cals**
4. then **Cal Sets**

◀ **Programming Commands** ▶



Calibration Selection dialog box help

This dialog box allows you to manage and apply Cal Sets.

Although the number of Cal Sets you can have is limited only by the amount of PNA memory, it is considered unusual to have more than about 10 existing Cal Sets, or one current Cal Set for every unique channel setup. Old Cal Sets (with 'stale' data) should be deleted or overwritten.

The active channel's Cal Register always appears, even if empty. Cal Registers that belong to other channels appear in the list of Cal Sets only if the channel exists, and only if they contain data.

- Learn about [Cal Registers](#).
- Learn how to [View the Error Terms of a Cal Set](#).

To apply a Cal Set to the active channel, click a row to select that Cal Set, then click Apply Cal.

Note: A Cal Set must have been generated from the same [measurement class](#) as the active channel in order for it to Applied.

Columns click a heading to sort by that column

Cal Set Name Name to identify the Cal Set.

Description User-settable text to further identify the Cal Set.

Channels Channel numbers that are currently using this Cal Set. A blank entry means it is not currently in use.

CalType / Ports Type of Cal contained in the Cal Set. [Learn about applying appropriate Cal Types](#).

Cal Type Abbreviations:

1P, 2P, 3P, 4P... - Full n-Port calibration

+ - Indicates a Power Correction is included in the Cal Set

R - Response (instead of ports, shows the measurement type that it corrects.)

ER/x-y [Enhanced Response](#), where **x** is the receive port; **y** is the source port.

VMC [Vector Mixer Cal](#)

SMC [Scalar Mixer Cal](#)

Modified Date and time the Cal Set was last modified.

Buttons

Copy Invokes the [Save as User Cal Set](#) dialog box. Type a name for the copy of the selected Cal Set data.

Show / Edit Properties Invokes the Cal Set Properties dialog box. This allows you to view all of the Cal Set properties and create a **duplicate** User Cal Set from an existing User Cal Set or Cal Register.

Delete Permanently deletes the Cal Set after you choose OK to a warning prompt.

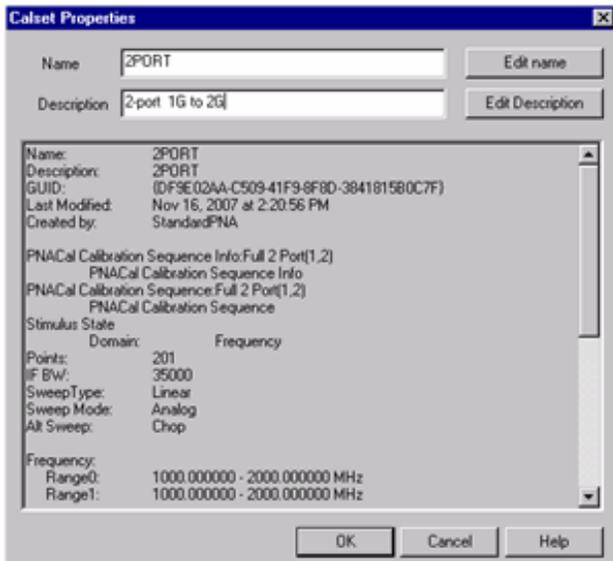
Delete All Permanently deletes ALL listed Cal Sets and Cal Registers after you choose OK to a warning prompt.

Apply Cal Applies the selected Cal Set to the active channel. If the stimulus settings of the Cal Set and channel are different, [a choice must be made](#).

Unselect Available ONLY if the selected Cal Set is being used by the active channel. Click to 'Un-apply' the Cal Set, then click **Close** to exit with the Cal Set un-applied.

OK Always APPLIES THE SELECTED CAL SET to the active channel, then closes the dialog box.

Close Exit the dialog box. Performs no further action.



Cal Set Properties dialog box help

Allows you to view all of the Cal Set properties and create a **duplicate** User Cal Set from an existing User Cal Set or Cal Register.

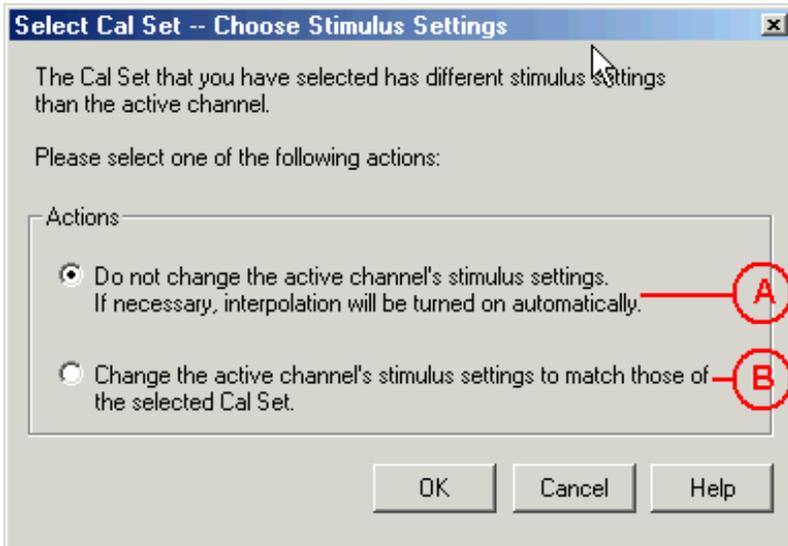
Name Edit name of the User Cal Set. You can NOT change the name of a Cal Register.

Description Descriptive text to further identify the Cal Set.

Cal Set Properties Lists descriptive information and stimulus conditions of the Cal Set.

Learn how to [View the Error Terms of a Cal Set.](#)

Stimulus Setting Different between Cal Set and Measurement



Select Cal Set -- Choose Stimulus Settings dialog box help

The Cal Set contains the channel stimulus settings that were in place when the Cal Set was saved. This dialog appears when the Cal Set channel settings are different than those of the channel to which the Cal Set is being applied. Choose between the following options.. (See above image).

- A. Keep the Active Channel Stimulus settings. Interpolate if possible.
 - If the Cal Set frequency range is greater the active channel, then Interpolation will be turned ON. Learn more about [Interpolation Accuracy](#)
 - If the Cal Set frequency range is less than the active channel, then this option is not available.
- B. Keep the Cal Set Stimulus settings. The Active Channel stimulus setting are changed.

OK Make the change.

Cancel Cal Set will NOT be applied.

Examples of Cal Set Usage

The following examples show how Cal Sets increase flexibility and speed in making analyzer measurements.

- Using one User Cal Set with many Channels
- [Using one Measurement with many Cal Sets](#)

Using one User Cal Set with many Channels

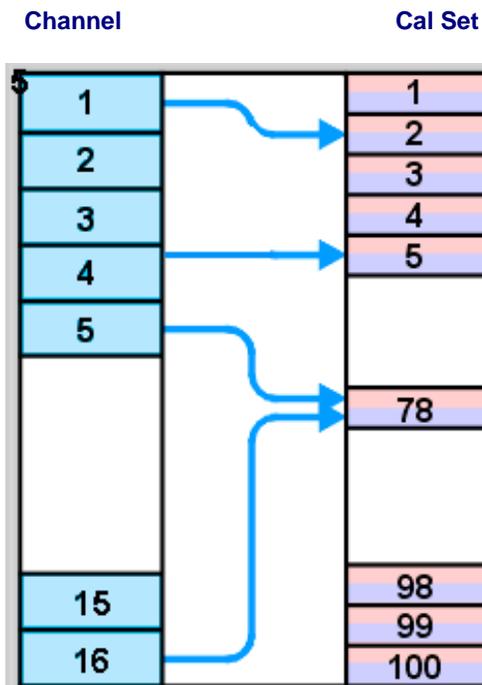
It is possible to do one calibration, then apply it to several channels.

An example:

During a manufacturing process, you may have many calibrated channels. You may wish to continuously cycle through the measurements and examine them individually. Occasionally, you may wish to refresh the calibration without having to recreate all the measurement state files.

Here is how: Examine the stimulus settings for each channel. Then make the User Cal Set stimulus range a super-set of the whole group. Each channel can then use the same User Cal Set. Some calibrations will be interpolated. **Note:** Make sure that [interpolation](#) is turned on.

Notice in the following image, Cal Set 78 is used on more than one channel, in this case Channel 5 and 16 .



Using one Measurement with many Cal Sets

The drawback with having one very large User Cal Set associated with many instrument states could be a loss of accuracy due to interpolation. In such cases, consider using one User Cal Set for each stimulus setting.. The stimulus conditions can then be changed for a channel by applying different User Cal Sets. Other settings (window setups, measurement definitions, scaling, limits, markers) will not change. This may result in faster state changes

than if you saved and recalled *.cst files for each set of stimulus conditions.

Example #1: An amplifier needs to be measured at several input power levels. Calibrate at several power levels and save each calibration in a separate User Cal Set. Then, apply the User Cal Sets to the single measurement consecutively.

Example #2: Making an S21 Measurement, you need to measure both wide span and narrow span characteristics of the device. One Cal Set covers the wide span setup; another the narrow span setup.

Archiving Cal Sets using .cal or .csa files

Because User Cal Sets can easily be deleted, provide extra backup by also saving your calibration as a .cal or .csa file ([see saving a .cal file](#)).

Example:

One person performs a calibration, names and saves it as a User Cal Set. This Cal Set is available for any other person to use. A second user could accidentally delete or modify the User Cal Set requiring the originator to repeat the calibration.

Security can be provided for calibration data by saving the Cal Set to a **.cal** file or **.csa** file. At a later time, the file could be recalled and the original calibration restored.

Last modified:

24-Jul-2012	Added + to cal type
17-Sep-2009	Edit wording in Appending...
3-Sep-2008	Removed legacy content
22-Apr-2008	Added link to UI Cal Set preference
9/12/06	Added link to programming commands

Error Correction and Interpolation

Error Correction and Interpolation settings work together to provide you with the highest level of calibration accuracy possible.

- [How to set Error Correction](#)
- [Error Correction](#)
- [Viewing Correction Levels](#)
- [How to set Interpolation](#)
- [Interpolation Accuracy](#)

See other Calibration Topics

How to set Error Correction

Using front-panel HARDKEY [softkey] buttons

1. Press **CAL**
2. then [**Correction ON/off**]

Using a mouse with PNA Menus

1. Click **Cal**
2. then **Correction ON/off**

[Programming Commands](#)

Error Correction

The Error Correction ON setting means that the calibration error terms are applied to the measurement. Error Correction is automatically turned ON when a calibration is performed or if a Cal Set is applied to a measurement. The PNA attempts to turn error correction ON for ALL of the measurements on the active channel. This may not always be possible when applying Cal Sets. For more information, see [Applying Cal Sets](#).

When full 2-port error correction is ON, both forward and reverse sweeps are required to gather all 12 error terms, even if only one reflection measurement is displayed. This may result in a higher measurement speed than expected. [Learn more](#).

You can always turn Error Correction OFF for the active measurement by clicking Correction OFF. The PNA will turn Error Correction OFF automatically when making stimulus changes [under some conditions](#). To turn correction back ON, click **Correction ON**. Then:

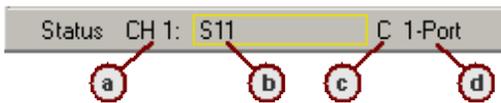
- If Interpolation can NOT be performed, a dialog box will ask if you would like to [change the stimulus settings](#) to those of the applied calibration. Click OK or Cancel.
- If Interpolation can be performed, the stimulus setting will change and correction turned ON.

Viewing Correction Level

The correction level provides information about the accuracy of the active measurement. Correction level notation is displayed on the status bar for different calibration types like response, full 2-port, TRL, or power calibration.

To View Correction Levels:

In the **View** menu, click **Status Bar**. The status bar appears and displays the following items:



- Active Channel
- Measurement parameter
- Correction Level (see description below)
- Calibration type

	Correction Level	Accuracy
C	Full	Highest
C*	Interpolated	Uncertain
CΔ	Changed	Uncertain
No Cor	No Correction	Lowest

C Full Correction

Full Correction level is displayed immediately after a calibration is performed or when a valid Cal Set is applied. If you require optimum accuracy, avoid adjusting analyzer settings after calibration so your measurement remains at this level.

C* Interpolated Correction

"C star" appears in the status bar when a measurement is being interpolated. See Interpolation (above) and [Interpolation Accuracy](#).

CΔ Changed Settings

"C-delta" appears in the status bar when one or more of the following stimulus settings change. The resulting measurement accuracy depends on which parameter has changed and how much it has changed. For optimum accuracy, recalibrate using the new settings.

- [Sweep time](#)
- [IF Bandwidth](#)
- [Port power](#)
- [Stepped sweep enabled/disabled](#)

No Corr No Correction

The following will cause the PNA to turn Error Correction OFF for the channel:

- Decrease the start frequency
- Increase the stop frequency
- Change start frequency, stop frequency, or number of points with Interpolation OFF.
- [Change sweep type](#)

How to set Interpolation

Using front-panel HARDKEY [softkey] buttons

1. Press **CAL**
2. then **[More]**
3. then **[Interplotion ON/off]**

Using a mouse with PNA Menus

1. Click **Cal**
2. then **More**
3. then **Interplotion ON/off**

◀ Programming Commands ▶

Interpolation

Calibration interpolation adjusts calibration error terms to match changes to the following settings that you make AFTER a calibration is performed or a [Cal Set applied](#).

The Interpolation **ON** setting means that interpolation is **enabled** for the active measurement. This does not necessarily mean that the measurement is interpolated. When enabled (ON), if interpolation becomes necessary because you change any of the following stimulus settings, **then** interpolation will

be applied. When stimulus settings change while interpolation is OFF, interpolation is NOT applied but instead, error correction is turned OFF.

Interpolation occurs (if enabled) when you change any of the following settings:

- Start frequency increased
- Stop frequency decreased
- Number of points

Note: Decreasing the start frequency, or increasing the stop frequency will always turn correction **OFF**. (Exception: [Power Calibration](#) DOES extrapolate to the start and stop frequencies.)

Interpolation Accuracy

When a measurement is interpolated, the accuracy of the measurements cannot be predicted. It may be affected significantly or not at all. Identifying measurement errors in these cases must be determined on a case-by-case basis. In general, the magnitude and phase stimulus from the PNA and the response from the DUT need to be smooth and continuous for measurement interpolation to give accurate results.

Significant measurement inaccuracy **WILL** occur when the phase shift response between measurement points increases changes more than 180 degrees. The PNA will incorrectly interpolate the new phase data. For more information, see [phase accuracy](#).

In general, the chances of significant inaccuracy increases when interpolating measurements under the following conditions:

- when frequency span between measurement points becomes much greater.
- when measurement frequencies are above 10 GHz where phase changes happen more rapidly.
- when interpolating across frequency band crossings. [Learn more about band crossings](#).

Note: When the interpolation algorithm encounters an abrupt or large change in the response magnitude or phase, such as can occur at band crossings, large interpolation errors can be included in the displayed data. These errors can be seen as steps or spikes. If this occurs, consider turning off interpolation, changing the measurement parameters, or creating [sweep segments](#) that skip over the band crossings.

Last modified:

19-Feb-2013 Added Accuracy note (SS)
3-Sep-2008 Removed legacy content
12-Sep-2006 Added link to programming commands

Using ECal

This topic discusses all aspects of performing an ECal:

- [ECal Overview](#)
- [Connect ECal Module to the PNA](#)
- [How to Perform a Calibration Using ECal](#)

See Also:

[ECal User-Characterization](#)

[Perform a 4-Port Cal with ONE 2-Port ECal Module](#)

[Restore ECal Module Memory](#)

[See other Calibration Topics](#)

ECal Overview

ECal is a complete solid-state calibration solution. Every ECal module contains electronic standards that are automatically switched into position during a PNA measurement calibration. These electronic standards have been measured at the factory and the data stored within the memory of the ECal module. The PNA uses this stored data, along with the PNA-measured data, to calculate the error terms for a measurement calibration.

ECal modules are available in 2-port and 4-port models and a variety of connector types, covering many frequency ranges. See [Analyzer Accessories](#) for more about available ECal modules and ordering information.

You can perform the following calibrations with ECal:

- 1-Port Reflection calibration
- Full 2-Port calibration
- Full 3-Port calibration
- Full 4-Port calibration

Verify the validity of a mechanical or ECal calibration with [ECal confidence check](#).

Care and Handling of ECal Modules

You can improve accuracy, repeatability, and avoid costly repair of equipment in the following ways.

- Practice proper connector care. See [Connector Care](#).
- Protect equipment against ESD damage. Read [Electrostatic Discharge Protection](#).

Power Level into an ECal module

- NEVER exceed the following Damage levels to the ECal module.
- For highest accuracy, do not exceed the following ECal Compression levels when calibrating:

Model	Compression level	Damage level
N469x series	-5 dBm	+10 dBm
N4432A series	-7 dBm	+20 dBm
N4433A series		
N4431x series	+7 dBm	+20 dBm
8509x series	+9 dBm	+20 dBm

When using the PNA-X, the power level can be increased after calibration with minimal impact on measurement accuracy.

Connect ECal Module to the PNA

ECal modules are controlled and powered through a USB connection to the PNA. When you connect the module, the PNA automatically recognizes the type of module, frequency range, and connector type.

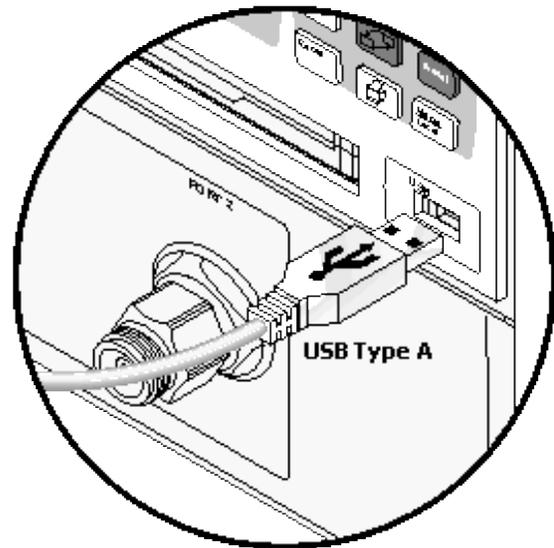
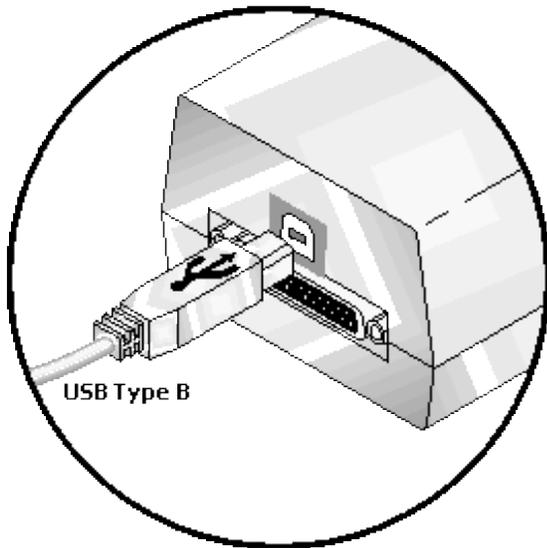
[See Important First-time USB connection note.](#)

ECal modules connect to the USB port on the front or rear panel of the PNA.

1. Wear a grounded wrist strap when making connections.
2. Connect the USB cable **Type B** connector to the ECal module and the USB cable **Type A** connector to the front or rear panel USB connector of the analyzer, as shown in the following graphics.

ECal Module USB Port

Analyzer Front Panel USB Port



Notes:

- Unused ECal modules that have completed a calibration may remain connected to the USB port.
- You can connect and disconnect the ECal module while the analyzer is operating. However, DO NOT connect or disconnect the module while data transfer is in progress. This can result in damage or at least corrupted data.

How to Perform a Calibration Using ECal

Select an ECal module that has connectors of the same type and gender as the DUT. If such an ECal module is not available, a module with connectors different from the DUT can be used by using [Advanced Settings](#) or [User Characterization](#). See Also: [Perform a 4-Port Cal with ONE 2-Port ECal Module](#)

Connect the ECal module ports to the PNA ports. During the calibration process the PNA can either automatically detect how the ECal module is connected, or the orientation can be performed manually.

1. Connect the ECal module USB cable to the analyzer USB. See [Connect ECal Module USB to PNA USB](#).
2. Allow the module to warm up until it indicates **READY**.
3. Enter the analyzer settings. See [Set Up Measurements](#).
4. Do one of the following to start the [Calibration Wizard](#)

**Using front-panel
HARDKEY [softkey] buttons**

Using a mouse with PNA Menus

1. Press **CAL**
2. then [**Start Cal**]
3. then [**Cal Wizard**]

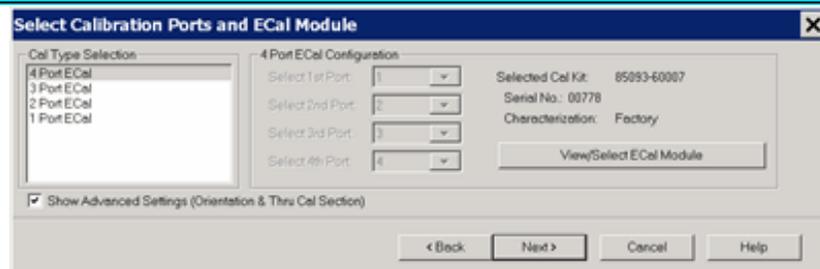
1. Click **Response**
2. then **Cal**
3. then **Cal Wizard**

Programming Commands

5. In the [Calibration Wizard Begin](#) dialog box, click **Use Electronic Cal (ECal)**.

Note: To calibrate with more than one ECal module, select [SmartCal](#), then choose the ECal modules as your Cal Kits.

Select Calibration Ports and ECal Module dialog box help



Allows you to select calibration type and settings.

Cal Type Selection / Configuration Select the number of ports to calibrate. Then select the port number configuration.

4 Port ECal Available only if using a 4-port PNA. No additional configuration necessary.

3 Port ECal Available only if using a 4-port or 3-port PNA.

2 Port ECal

1 Port ECal- (Reflection) Advanced Settings are not available.

View/Select ECal Module Click to [Select the ECal module](#) if more than one ECal module is connected to the PNA. Also, [Select the User Characterization](#) within the module. Learn more about [User Characterization](#).

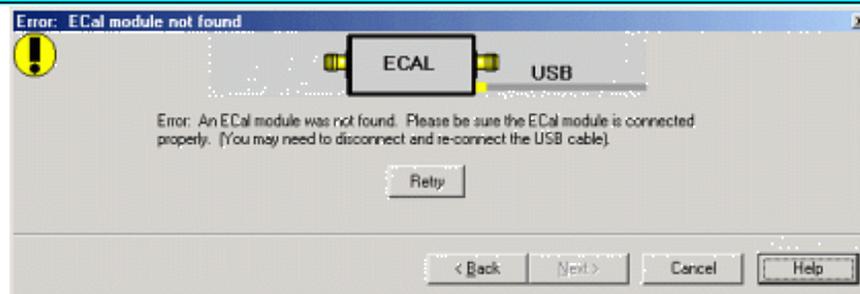
Show Advanced Settings Check to display the [Advanced Settings](#) when **Next** is clicked.

Back Return to [Cal Wizard Begin](#) dialog. If checked, you can clear the [Save Preferences](#) checkbox to see the Begin page when the Cal Wizard begins.

Note: The PNA no longer allows ECal isolation to be performed. The inherent isolation of the PNA is better than that attained with correction using an ECal module.

Terminate any unused ECal ports with a 50 ohm load.

ECal module not found dialog box help



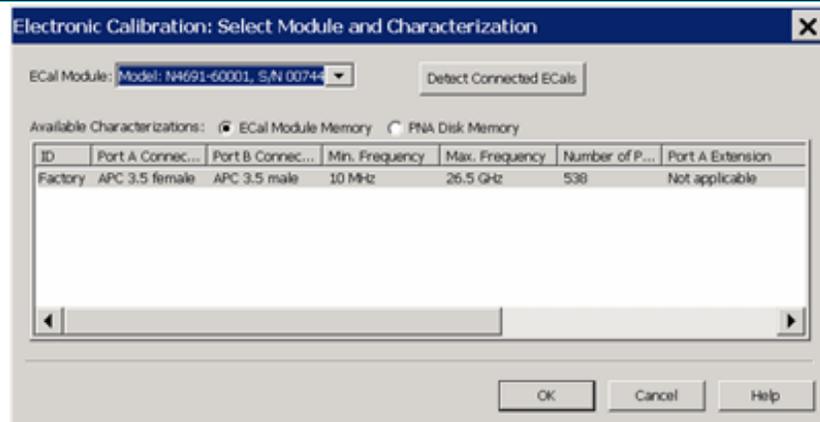
When this dialog appears, the ECal module is not connected or has not been recognized by the network analyzer.

Retry Check the USB connections and click to continue.

Notes:

- If your ECal module is not detected, try to unplug, then reconnect the USB connector to the PNA.
- When the ECal module is connected to the network analyzer for the first time, it may take approximately 30 seconds for the analyzer to recognize the module and make it available for calibration.
- For best accuracy, allow the ECal module to warm-up until it indicates READY.
- Agilent 8509x and N4431 ECal modules, when first connected, draw significantly more current than other modules. This could cause the USB to stop working in certain situations. [See USB limitations.](#)
- See [Connect ECal Module USB to PNA USB.](#)

Select Module and Characterization dialog box help



ECal Module Select one of the ECal modules that are connected to the PNA.

Detect Connected ECals Click to rescan the USB for ECal modules.

Available Characterizations

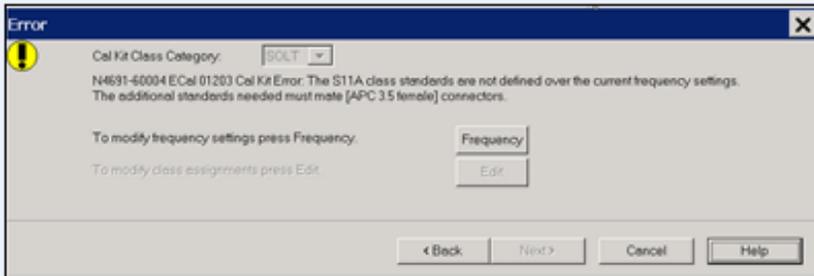
ECal Module Memory - Displays the factory and [user characterizations](#) that are stored in the ECal module.

PNA Disk Memory - Displays the user characterizations that are stored in PNA Disk Memory.

[Learn more User Characterizations in PNA Disk Memory.](#)

Select either the characterization data to use for the calibration. Once selected, that characterization becomes the default selection until the PNA is turned OFF and restarted. When restarted, **Factory** again becomes the default selection.

Error: Frequency Range dialog box help



When this dialog appears, the current cal standards (or ECal module) does not cover the current frequency range of the measurement. Do one of the following to correct the problem:

Cal Kit Class Category Not available with ECal modules.

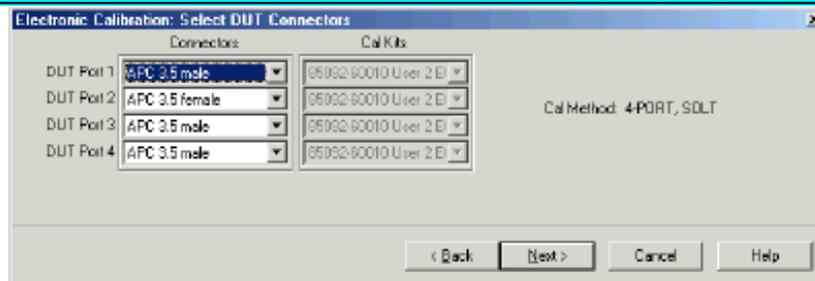
Frequency Change the frequency range of the active channel.

Edit Not available with ECal modules.

Back Select a different characterization that covers the required frequency range.

Cancel Re-characterize the module with an increased frequency range.

Select DUT Connectors and Cal Kits dialog box help

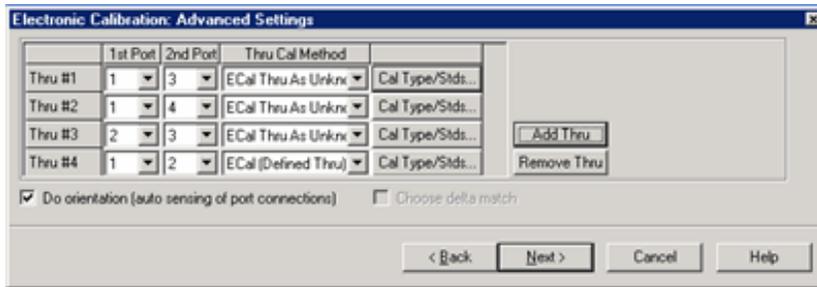


If the ECal module or selected User Characterization has more than one connector type, then the following dialog box is presented which allows you to describe the DUT connector type. Otherwise, click next to proceed to [Advanced Settings](#) (if checked) or [ECal Steps](#).

Connectors

The available connectors are listed for each DUT port.

Advanced Settings dialog box help



Thru #n

Lists the proposed Thru connections to be made during the calibration process. You can change these Thru connections to better suit your test setup.

- The proposed Thru connections are listed automatically.
- Additional Thru connections can be selected for higher accuracy. [Learn more](#).
- For Balanced measurements, [learn which Thru paths to select](#).

Add Thru

Click to add a Thru connection. [Learn more](#)

Remove Thru

Select a Thru by clicking the "Thru #N" field or the "1st Port / 2nd Port" field. Then click "Remove Thru". This selection is NOT available if the selected Thru is required for the calibration.

1st Port / 2nd Port

Click to change the two ports to be included in the Thru connection. The order of the port numbers (1st or 2nd) is not critical.

Thru Cal Method

Lists the available Thru Cal methods for the specified port pairs.

[Learn about ECal Thru Methods](#)

Cal Type/ Std

Click to invoke the [View / Modify Properties of Cal dialog box](#)

Do orientation

When this box is checked (the default setting) the PNA automatically senses the model and

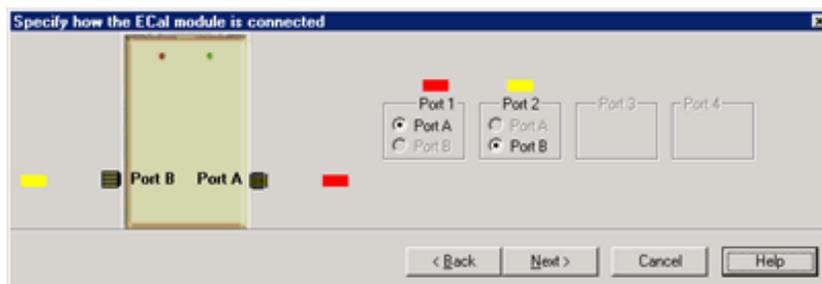
direction in which an ECal module port is connected to the PNA ports. If power to the ECal module is too low, it will appear as if there is no ECal module connected. If you use low power and are having this problem, clear this check box to provide the orientation manually.

Orientation occurs first at the middle of the frequency range that you are calibrating. If a signal is not detected, it tries again at the lowest frequency in the range.

Choose delta match

Available only when a Delta Match Cal is required.

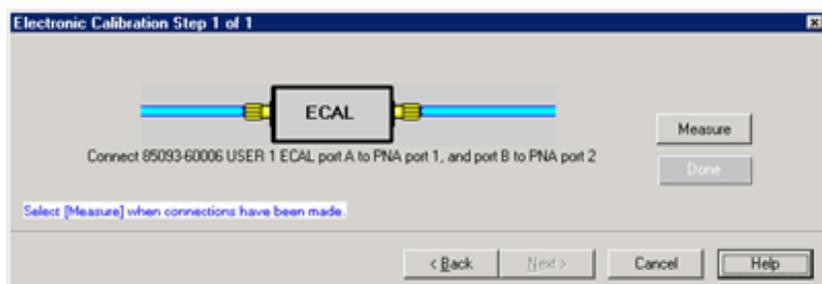
- Check, then click **Next** to invoke the [Select Cal Set for Delta Match](#) dialog box.
- Clear - The Cal Wizard uses the [Global Delta Match Cal](#) if available.



Specify how the ECal module is connected dialog box help

This dialog box appears when the **Do orientation** checkbox in the previous dialog box is cleared.

Click the ECal Port that is connected to each PNA port.



Electronic Calibration Steps dialog box help

Note: Beginning in PNA Rev. 6.0, ECal can be performed with External triggers. [Learn more.](#)

Displays the instructions for each measurement required for calibration.

Measure Measures the ECal standards.

Done Click when last standard has been measured.

Saving an ECal Calibration

When complete, you can save the new calibration. [Learn how.](#)

Last modified:

2-Aug-2011	Added 4-port Cal with 2-port ECal
28-Jun-2011	Added link to first time note
9-Jan-2009	Updated compression level
3-Sep-2008	Removed legacy content
27-Jun-2008	Added compression levels
4-Sep-2007	Added First time note
14-Sep-2007	MX Added UI
Sept. 12, 2006	MQ Modified images for multiport

ECal User Characterization

- [Overview](#)
- [How to Perform a User Characterization](#)
- [Manage Disk Memory](#)
- [Restore ECal Module Memory](#)

See Also

[Using ECal](#)

[Perform a 4-Port Cal with a 2-Port ECal Module](#)

Other Calibration Topics

Overview

A user-characterized ECal module allows you to add adapters to the ECal module, re-measure the standards in the ECal module, INCLUDING the adapters, then add that data to ECal memory or save it to the PNA hard drive. This extends the reference plane from the module test ports to the adapters.

Why perform a User Characterization?

- If you need to use adapters with your ECal module, you could characterize your ECal module with the adapters attached and perform subsequent ECals in a single step.
- If you have a 4-port ECal module, you could configure the module with adapters of different connector types, then perform a User Characterization of the module. When you need to test a DUT with a pair of the connector types on your module, calibrate the PNA with a 1-step ECal using the same two connectors on the User-characterized module.
- If you test devices in a fixture, you could embed the characterization of the fixture in the characterization of the module. To do this, during the mechanical calibration portion of the User Characterization, calibrate at the reference plane of the device as you would normally calibrate. Then remove the fixturing to be embedded and insert the ECal module to be characterized. When measuring the ECal module, the PNA removes the effects of the fixturing and stores the measurement results in the user characterized ECal module. Subsequent calibrations with that user-characterized module will also remove the fixture effects.

Notes:

- Both 2-port and 4-port ECal modules support User Characterization.

- User Characterization does not delete the factory characterization data. The factory data is saved in the ECal module in addition to the User Characterization data.
- The ECal Data Wipe Utility is the only way that data can be deleted from the module. Learn more at <http://na.tm.agilent.com/pna/apps/applications.htm>.
- Beginning with A.06.03, a User Characterization can be performed beyond the frequency range of the ECal module. Although this practice is allowed, calibration accuracy with the extended User Characterization is likely to be degraded. To determine the level of degradation, compare measurements of a variety of devices using a PNA with a mechanical cal kit calibration versus an ECal extended User Characterization calibration.
- Beginning with A.08.33, you can save up to 12 User Characterizations in a single ECal module. Previous releases allowed up to 5. There are memory limitations. The PNA will determine if the contents of a User Characterization will fit inside the module before it is performed.
- Saving a new User Characterization with PNA releases **before** A.08.33 will erase any User Characterizations greater than 5. This is because the PNA completely rewrites ALL ECal User Characterizations when saving a new one in order to free storage space in the ECal module.
- Beginning with A.08.33, a User Characterization can be performed remotely. [See programming commands.](#)

User Characterizations can be saved to the **PNA Disk Memory** beginning with A.09.00. [Learn how.](#) Previously, User Characterizations could be saved only to the ECal module memory.

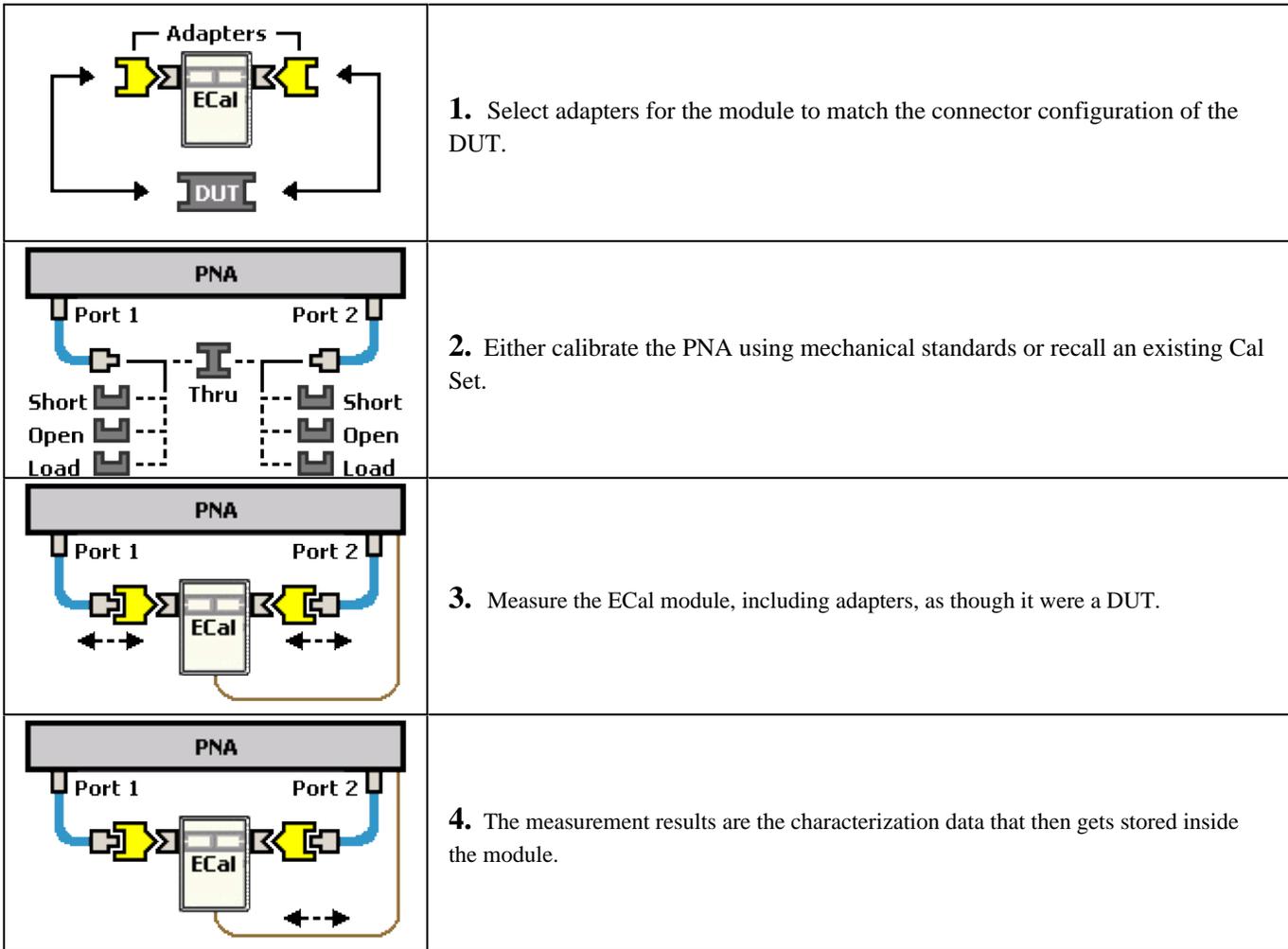
This feature provides the following benefits:

- A User Characterization using connectors that are NOT included in the [supported connector table](#) can NOT be stored to the ECal module. But when stored to disk memory, ANY connector type is allowed by firmware using a description of any length for the User Characterization.
- There is NO limit to the number of data points allowed in a User Characterization stored to disk memory. When stored in the ECal module, the number of data points is limited by the PNA firmware.
- The number of User Characterizations that can be stored to disk memory is limited only by available disk space.
- User Characterizations stored to disk memory can be freely shared between PNAs that have PNA Rev. 9.00 or later.

[Learn how to Manage User Characterization in Disk Memory.](#)

How to Perform a User Characterization

SUMMARY (A [detailed procedure](#) follows.)



Note

A 2-port PNA can be used to perform a User Characterization on a 4-port ECal module.

However, a 4-port ECal module has SIX different port pairs. The PNA must be recalibrated for each port pair that uses unique connector types or gender.

- If all 4 ECal module ports have the same connector type and gender, then only one PNA calibration is required to measure all six port pairs.
- If all 4 ECal module ports have different connector types or gender, then 6 calibrations are required.

When more than one PNA calibration is required during a User Characterization, then ALL calibrations must be performed using the standard Cal Wizard, saved to Cal Sets, and then [recalled from Cal Sets](#) DURING the User Characterization.

Detailed steps to Perform a User Characterization

1. Connect the ECal module to the network analyzer with the USB cable. See [Connect ECal Module USB to PNA USB](#).
2. Allow the module to warm up until it indicates **READY**.
3. **Preset** the analyzer.
4. [Set up the measurement](#). For best accuracy, the **IF bandwidth** should be set to **1 kHz** or less.
5. Start and complete the **Characterize ECal Module** Wizard:

Using front-panel HARDKEY [softkey] buttons

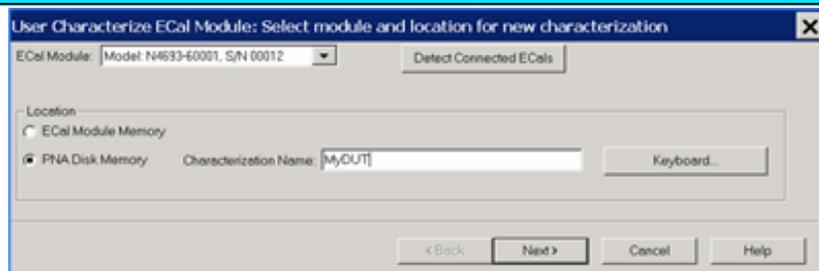
1. Press **CAL**
2. then **[More]**
3. then **[ECal]**
4. then **[Characterize ECal Module]**

Using a mouse with PNA Menus

1. Click **Response**
2. the **Cal**
3. then **More**
4. then **ECal**
5. then **Characterize ECal Module**

Programming Commands

Select Module and Location dialog box help



ECal Module Select one of the ECal modules that are connected to the PNA.

Detect Connected ECals Click to rescan the USB for ECal modules.

Location

- **ECal Module Memory** Click Next to see the following dialog.

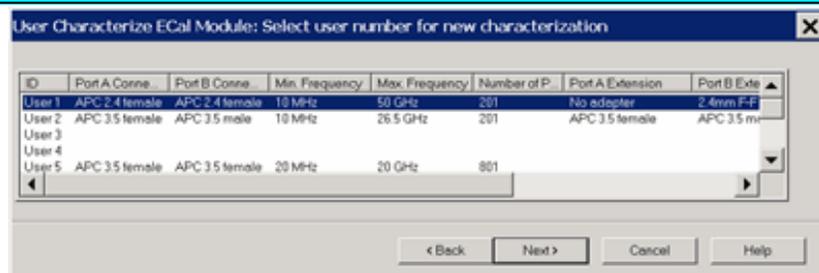
- **PNA Disk Memory** Enter a Characterization Name. This name appears when selecting a User Characterization to be used with subsequent calibrations.
 - [Learn how to manage characterizations that are stored to PNA disk memory.](#)
 - [See the benefits of storing the User Characterization to PNA Disk Memory.](#)

Keyboard Launches a keypad that can be used to type a characterization name from the PNA front panel.

Next Click to continue to the [Select Connectors for the Characterization](#) dialog box.

[See note regarding extended frequency use.](#)

Select User Number for new characterization dialog box help

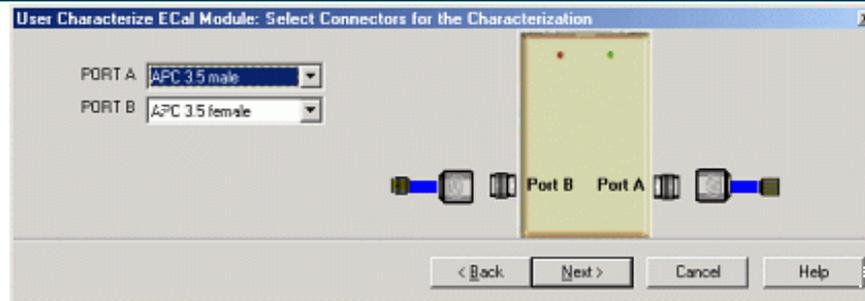


Scroll to view all of the parameters of the stored characterizations. Select an empty location or select to overwrite an existing characterization.

Next Click to continue to the [Select Connectors for the Characterization](#) dialog box.

[See note regarding extended frequency use.](#)

Select Connectors for the Characterization dialog box help



Connector Notes

When performing an ECal User Characterization, do NOT use a [custom connector name](#) that you added to this list. If you need to use a custom-defined connector type, select "Type B", or one of the "Type A" variations from the list of connectors for each port.

A User Characterization using connectors that are NOT included in the [supported connector table](#) can NOT be stored to the ECal module. But when stored to disk memory, ANY connector type is allowed. [Learn more about storing to PNA Disk Memory.](#)

Select the adapters for the ECal module test ports. Select **No adapter** if no adapter is used on a port.

PORT A Lists the connector types available for Port A.

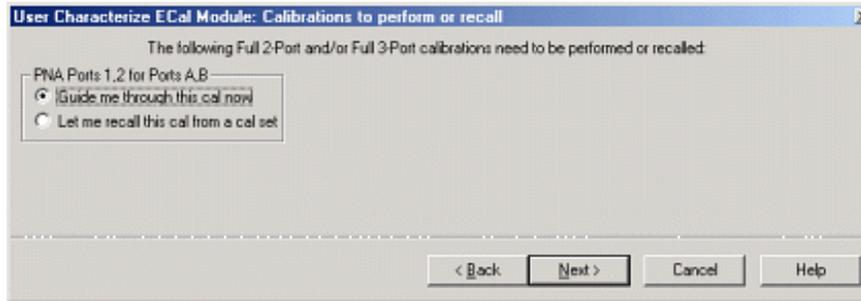
PORT B Lists the connector types available for Port B.

PORT C Lists the connector types available for Port C (available with a 4-port ECal module).

PORT D Lists the connector types available for Port D (available with a 4-port ECal module).

Next Click to continue to the [Calibrations to perform or recall](#) dialog box.

Calibrations to perform or recall dialog box help



The PNA must be calibrated before measuring the ECal module and necessary adapters. This dialog box displays the number and types of mechanical calibrations required for the characterization.

Guide me through this cal now Click to perform a Guided calibration. A calibration kit is required for each connector type.

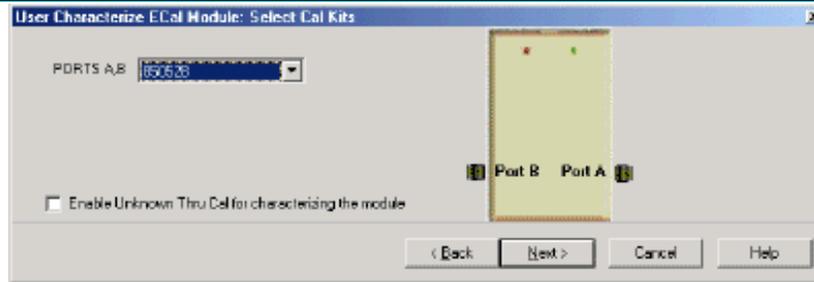
Note: Some PNA-L models cannot perform TRL calibration during the calibration portion of a User Characterization. However, this type of Cal can be performed using the Cal Wizard, saved to a Cal Set, then recalled at this point in the User Characterization.

If more than one calibration is required, this selection is not available. [See Note.](#)

Let me recall this cal from a cal set Click to select an existing Cal Set. You cannot select a Cal Set that is currently in use. Learn more about [Using Cal Sets.](#)

Next Click to continue to either the [Select Cal Kits](#) or the [Select Cal Set](#) dialog box.

Select Cal Kits dialog box help

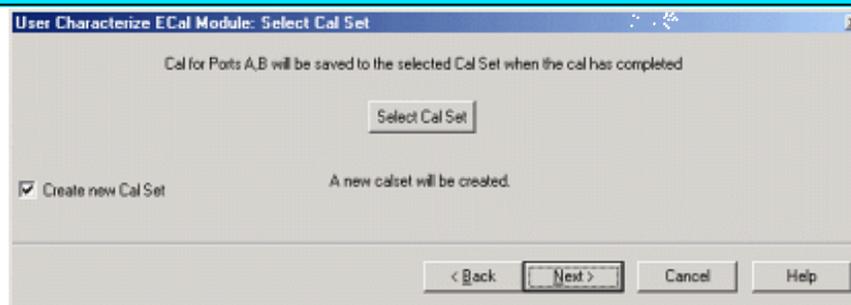


Provides a list of calibration kits to perform the calibration. Select the Cal Kit you will use for each port.

Enable Unknown Thru for characterizing the module Check to enable. This reduces the number of steps required to characterize the THRU standard. This setting is available only on PNA models with [one reference receiver per test port](#).

Next Click to continue to the [Select Cal Set](#) dialog box.

Select Cal Set dialog box help



The calibration that you perform will be written to a Cal Set. This dialog box allows you to select a Cal Set to overwrite, or to write to a new Cal Set. The current choice is visible below the **Select Cal Set** button.

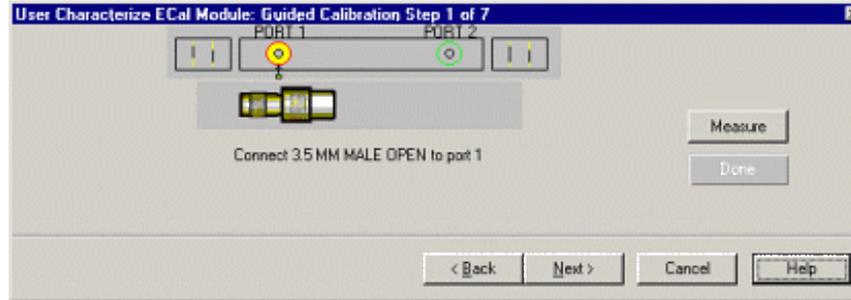
Select Cal Set Click to open the **Select A Cal Set** dialog box.

Create new Cal Set Check to create a new Cal Set to store the calibration. Clear to select and overwrite a stored Cal Set.

Next Click to continue to the [Guided Calibration Steps](#) dialog box.

Note: Remember the Cal Set name for future reference.

Guided Calibration Steps dialog box help



Instructs you to connect each calibration standard to the measurement port.

Measure Click to measure the standard.

Back Click to repeat one or more calibration steps.

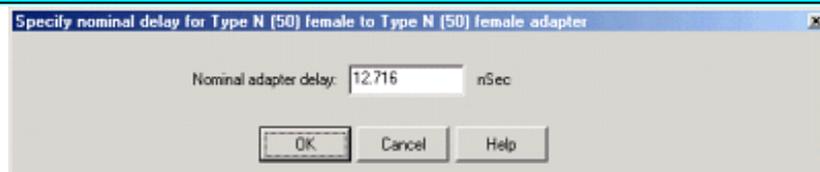
Done Click **after** a standard is re-measured and all measurements for the calibration are complete.

Next Click to continue to the next calibration step. (Does **not** measure the standard.)

Cancel Exits Calibration Wizard.

The **Specify nominal delay** or **Guided Calibration completed** dialog box appears when the steps are completed.

Specify nominal delay dialog box help



This dialog **ONLY** appears when [Adapter Removal](#) or [Unknown Thru](#) calibrations are performed.

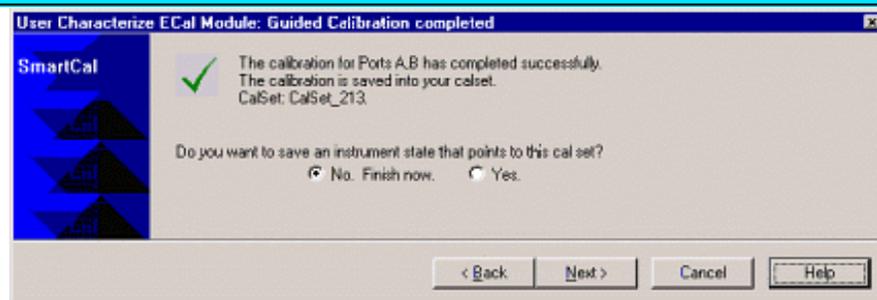
The following values were estimated from the measurement. Most of the time, they are adequate. However, for CW sweep or frequency sweep with large step sizes, the accuracy of the values may be improved.

Nominal adapter delay To improve this value, measure and record the delay of the adapter with a dense step size. Enter that value here.

Nominal phase offset (Waveguide ONLY). To improve this value, measure and record the phase offset of the Waveguide adapter with dense step size. Enter that value here.

When one connector is coax and the other connector is waveguide, the phase offset has an ambiguity of 180 degrees. For consistency, the estimate provided here is always between 0 and 180 degrees. You can change this estimate to any value between -180 degrees and +180 degrees.

Guided Calibration completed dialog box help



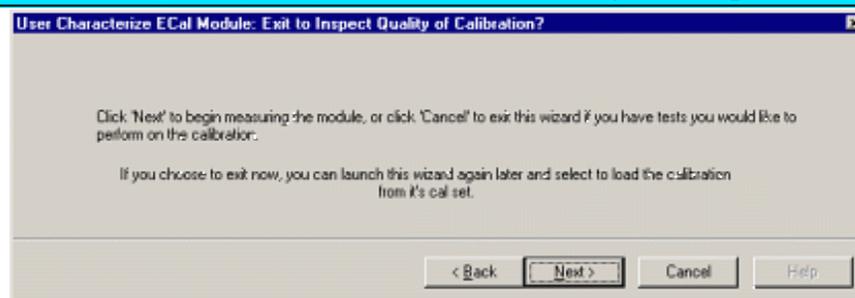
Allows you to finish the calibration and continue to the next characterization steps.

No. Finish now Select to save Cal Set data.

Yes Allows selection of Save options.

Next Click to continue to the [Exit to Inspect Quality of Calibration](#) dialog box.

Exit to Inspect Quality of Calibration dialog box help



Allows you to exit User Characterization to [validate the calibration](#) before proceeding with the characterization.

Back Allows you to repeat calibration.

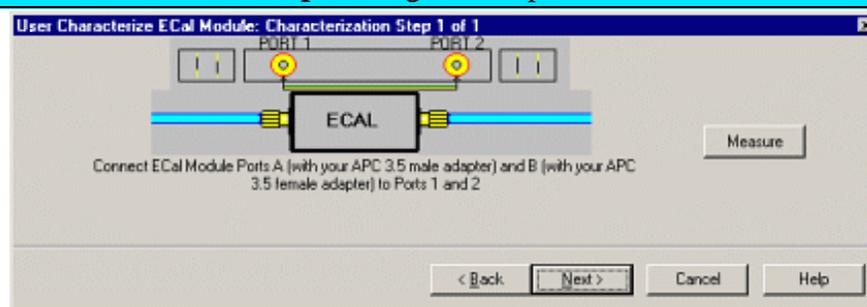
Next Click to continue to the [Characterization Steps](#) dialog box.

Cancel Exits the Calibration.

To return to the current step:

1. Start User Characterization.
2. In the **Select user number for new characterization** dialog box, click **Next**.
3. In the **Select Connectors for Characterization** dialog box, click **Next**. (Previous entry is stored in memory.)
4. In the **Calibrations to perform or recall** dialog box, recall the Cal Set that you just performed.

Characterization Steps dialog box help

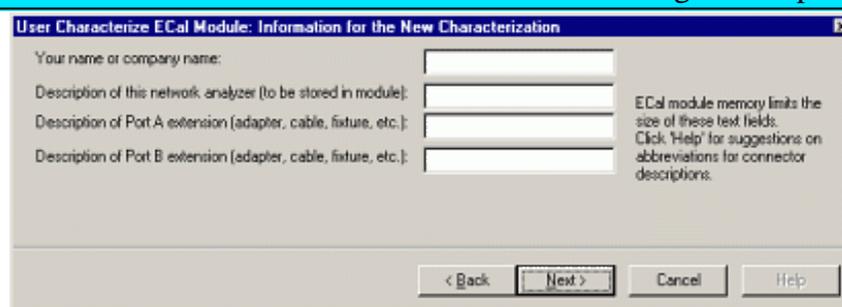


Describes the instructions for each measurement required for characterization.

Measure Measures the ECal module.

Next Click to continue to the [Information for the New Characterization](#) dialog box when measurements are complete.

Information for the New Characterization dialog box help



Allows you to describe the properties of the User Characterization.

Suggestions for connector abbreviations

To minimize the number of characters, we suggest using the following 3-character codes to describe the connectors listed.

A User Characterization using connectors that are NOT included on this list can NOT be stored to the ECal module. But when stored to disk memory, ANY connector type is allowed. [Learn more about storing to PNA Disk Memory.](#)

Connector Type	3-Character Code
1.0 mm female	10F
1.0 mm male	10M
1.85 mm female	18F
1.85 mm male	18M
2.4 mm female	24F
2.4 mm male	24M
2.92 mm female	29F
2.92 mm male	29M
3.5 mm female	35F
3.5 mm male	35M
7-16 female	16F
7-16 male	16M
Type F female	F7F
Type F male	F7M
N50 female	N5F
N50 male	N5M
N75 female	N7F
N75 male	N7M
APC 7	7MM
K-band waveguide	KBW
P-band waveguide	PBW
Q-band waveguide	QBW
R-band waveguide	RBW
U-band waveguide	UBW
V-band waveguide	VBW
W-band waveguide	WBW
X-band waveguide	XBW

Next Click to continue to the [Write Characterized Data to the ECal module](#) dialog box.

Write Characterized Data dialog box help



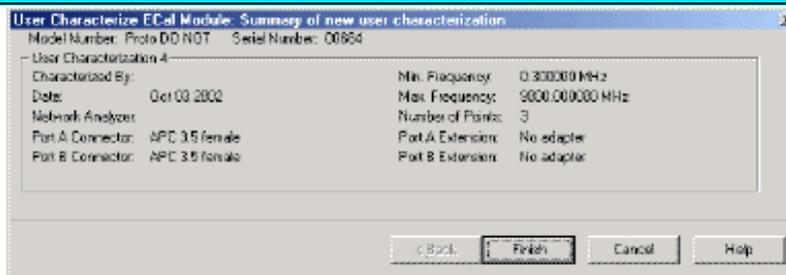
The PNA writes User Characterization and factory characterization data to either the PNA disk memory or the ECal module memory.

Write Click to write data.

The [Summary of new User Characterization](#) dialog box opens after data is saved to module.

- Existing data will be overwritten if you selected a User Characterization number that already has data. [Learn more](#)
- For more information, see [Restore ECal module memory](#).
- The ECal Data Wipe Utility is the only way that data can be deleted from the module. Learn more at <http://na.tm.agilent.com/pna/apps/applications.htm>.

Summary of new User Characterization dialog box help



Verify the status of the ECal User Characterization.

- ECal module model number

- summary from User Characterization

Cancel Click to exit (characterization complete).

Finish Click to exit (characterization complete).

Manage ECal User Characterizations in Disk Memory

Normally, User Characterizations that are stored in PNA disk memory can be used indefinitely without needing them to be managed. However, this dialog allows you to backup the characterizations in case they are accidentally erased, or to save them to a file that can be moved to another PNA.

How to Manage ECal User Characterizations in Disk Memory

Using front-panel HARDKEY [softkey] buttons

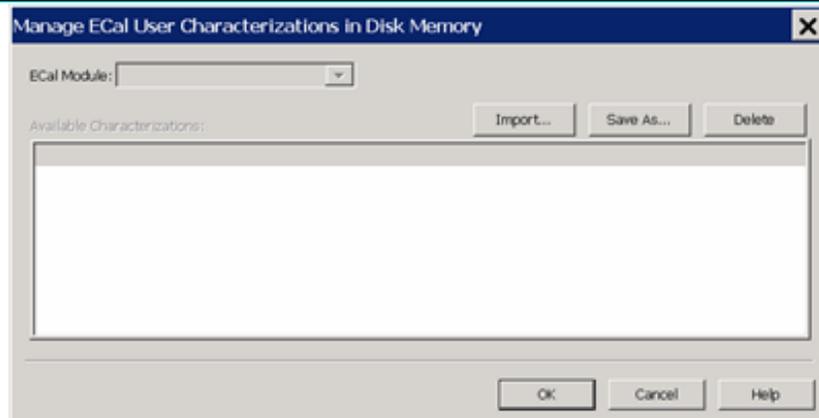
1. Press **CAL**
2. then **[More]**
3. then **[ECal]**
4. then **[Manage ECal Disk Memory]**

Using a mouse with PNA Menus

1. Click **Response**
2. then **Cal**
3. then **More**
4. then **ECal**
5. then **Manage ECal Disk Memory**

◀ Programming Commands ▶

Manage ECal User Characterizations in Disk Memory dialog box help



This dialog allows you to do either of the following:

- Save an existing User Characterization in disk memory to an *.euc file.
- Load a previously saved *.euc file for use on the PNA with the specified ECal module.

[Learn more about User Characterizations stored to Disk Memory.](#)

ECal Module Select an ECal Module from the list for which User Characterizations are currently stored in PNA disk memory.

Save As Saves a User Characterization that is currently in PNA disk memory to a *.euc file. This file can be used as a backup in case the archive file is accidentally deleted, or allows you to move the file to another PNA to be used with the selected ECal Module.

Import Loads a previously saved *.euc file for use on the PNA with the specified ECal module.

Delete Removes a User Characterization from PNA disk memory.

Restore ECal Module Memory

When user-characterized data is written to the ECal module, the entire contents of ECal memory is also written to the PNA hard drive, including the factory ECal data. In the unlikely event that your ECal module memory is lost, you can restore all ECal data to ECal memory.

Caution: If a new factory cal was performed **after** the ECal memory was written to the PNA, the new factory cal data will also be overwritten.

Note: An ECal Data Wipe Utility destroys all user data per US DoD 5220.22-M. Learn more at <http://na.tm.agilent.com/pna/apps/applications.htm>

How to Restore ECal Module Memory

Using front-panel HARDKEY [softkey] buttons

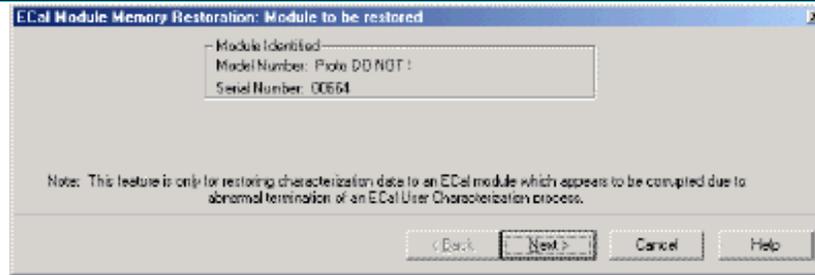
1. Press **SYSTEM**
2. then **[Service]**
3. then **[Utilities]**
4. then **[Restore...]**

Using a mouse with PNA Menus

1. Click **Utility**
2. then **System**
3. then **Service**
4. then **Utilities**
5. then **Restore...**

No Programming commands are available for this feature.

Module to be restored dialog box help



Verify the serial number of the module to be restored. If two modules are connected to the PNA , choose the one to have data restored.

Next Click to write data to the module.

Last modified:

7-Mar-2013	Minor edit to TRL cal note
10-Oct-2012	Edit suggested list of connector abrevs.
7-Jun-2012	Minor edits
15-Mar-2011	Added benefits of Disk Memory
22-Oct-2010	Clarified data delete
24-Aug-2009	Added PNA Disk Memory (9.0)
3-Nov-2008	Added 12 chars and remote link (8.33)
3-Sep-2008	Removed legacy content
28-Aug-2008	Clarified note about restore
13-Aug-2008	Added note about data wipe
25-Apr-2007	Added note about can NOT delete data.
20-Sept-2006	MX Modified for cross-browser

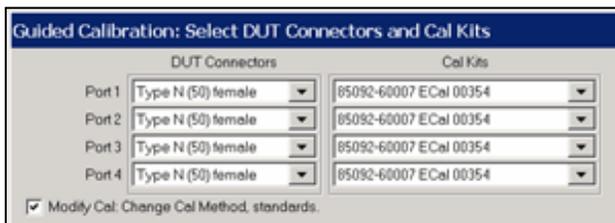
Perform a 4-Port Cal with One 2-Port ECal Module

You can perform a 4-port calibration with a 2-Port ECal Module. When all four DUT connectors are the same type and gender, the calibration can occur with only four connections, the same number of connections you would make with a 4-port ECal module.

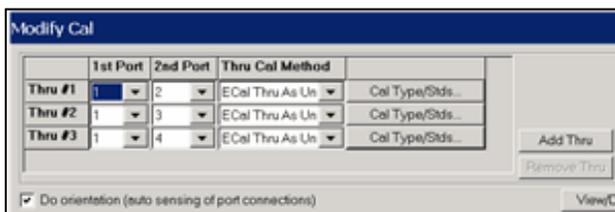
- The ECal module must span the frequency range of the measurement.
- The ECal module must have connectors that match the DUT connectors. Because we are using a 2-port ECal module, this means that the DUT must have only TWO unique connector types and gender. When the DUT has more than two connector types/genders, you can select a different cal kit for each port using SmartCal.

Procedure

1. Connect the 2-port ECal module to a PNA USB port.
2. Press **Cal**, then **Start Cal**, then **Cal Wizard**
3. Select either **SmartCal** or **Use Electronic Calibration** then click **Next**.
4. Select **4 Port Cal**, then click **Next** to see the following dialog:



5. Select the DUT Connectors for each port. In this example, all four DUT connectors are Type N, female.
6. Select the attached ECal module. We are using a **85092-60007 ECal** module.
7. Select **Modify Cal (Show Advanced Settings for ECal)** then click **Next** to see the following dialog:



8. For the fewest number of physical connections, select the default port assignments.
 - The **1st Port** selection for each port pair is 1.

- For single-ended (standard) measurements, THREE is the minimum number of Thru connections. For Balanced measurements, FOUR Thru connections should be made. [Learn more.](#)
 - For higher accuracy, select **Add Thru**. The Cal Wizard will add another port pair which results in more physical connections.
9. Select **ECal Thru as Unknown**. This is the most accurate and easiest Thru Cal Method. [Learn more.](#)
 10. You may need to clear **Do Orientation** when calibrating at low power levels. [Learn more.](#) This will add additional connection steps.
 11. Follow the prompts to complete the calibration:
 1. Connect ECal to ports 1 and 2. Click **Measure**.
 2. Connect ECal to ports 1 and 3. Click **Measure**.
 3. Connect ECal to ports 1 and 4. Click **Measure**.
 12. At the **Specify Delay** dialogs, click **OK**. This is the measured delay for each of the Thru connections in the ECal module. [Learn more.](#)
 13. Click either **Save As User Cal Set**, or **Finish**.
-

Last modified:

2-Aug-2011 New topic

TRL Calibration

TRL (Thru, Reflect, Line) represents a **family** of calibration techniques that measure two transmission standards and one reflection standard to determine the 2-port 12-term error coefficients. For example, **TRM** (Thru, Reflect, Match), **LRL** (Line, Reflect, Line), **LRM** (Line, Reflect, Match) are all included in this family.

The traditional SOLT calibration measures one transmission standard (T) and three reflection standards (SOL) to determine the same error coefficients.

- [Why Perform a TRL Cal?](#)
- [The TRL Calibration Process](#)
- [TRL Cal Kits](#)
- [Cal Standards Used in TRL](#)
- [TRL on 4-port PNA-L and ALL Models with an External Test Set](#)

See other Calibration Topics

Why Perform a TRL Cal?

TRL calibration is extremely accurate, in most cases more accurate than an SOLT cal. However, very few calibration kits contain TRL standards. TRL Cal is most often performed when you require a high level of accuracy and do not have calibration standards in the same connector type as your DUT. This is usually the case when using test fixtures, or making on-wafer measurements with probes. Therefore, in some cases you must construct and characterize standards in the same media type as your DUT configuration. It is easier to manufacture and characterize three TRL standards than the four SOLT standards.

Another advantage of TRL calibration is that the TRL standards need not be defined as completely and accurately as the SOLT standards. While SOLT standards are completely characterized and stored as the standard definition, TRL standards are modeled, and not completely characterized. However, TRL cal accuracy is directly proportional to the quality and repeatability of the TRL standards. Physical discontinuities, such as bends in the transmission lines and beads in coaxial structures, will degrade the TRL calibration. The connectors must be clean and allow repeatable connections.

To learn more about Cal Standard requirements, see [Cal Standards Used in TRL](#).

The TRL Cal Process

Although TRL can be performed using the Cal Wizard Unguided Cal selection, the following process uses the easier [SmartCal](#) selection. Both selections require that you already have TRL calibration standards defined and included in a PNA cal kit.

1. Preset the PNA
2. Set up a measurement and the desired stimulus settings.
3. Click **Calibration / Calibration Wizard**
4. Click **SmartCal (Guided Cal)**.
5. [Select the DUT connectors and Cal Kit](#) for each port. The LOWEST port number of each [port pair](#) MUST include TRL standards. TRL appears as the Cal Method.
6. Check **Modify Cal, Next**, then **View/Modify** to change [default TRL options](#) if necessary.
7. Follow the prompts to complete the calibration.
8. [Check the accuracy](#) of the calibration

TRL Cal Kits

Agilent Technologies offers two cal kits that include the required standards to perform a TRL calibration: 85050C (APC 7mm) and 85052C (3.5mm). Both kits include the traditional Short, Open, and Load standards. (The Thru standard, not actually supplied, assumes a [zero-length Thru](#)). In addition, the kits include an airline which is used as the LINE standard. To use the airline, the kits include an airline body, center conductor, and insertion / extraction tools. The APC 7 kit includes an adapter to connect the airline to the APC connector.

Cal Standards Used in TRL

These standards must be defined in your TRL cal kit:

THRU

Note: All [THRU calibration methods](#) are supported in a TRL Cal **EXCEPT** Unknown Thru.

- The THRU standard can be either a zero-length or non-zero length. However, a zero-length THRU is more accurate because it has zero loss and no reflections, by definition.
- The THRU standard cannot be the same electrical length as the LINE standard.
- If the insertion phase and electrical length are well-defined, the THRU standard may be used to [set the reference plane](#).
- Characteristic impedance of the THRU and LINE standards defines the reference impedance of the

calibration.

- If a THRU standard with the correct connectors is NOT available, an adapter removal cal can be performed.

REFLECT

- The REFLECT standard can be anything with a high reflection, as long as it is the same when connected to both PNA ports.
- The actual magnitude of the reflection need not be known.
- The phase of the reflection standard must be known within 1/4 wavelength.
- If the magnitude and phase of the reflection standard are well-defined, the standard may be used to [set the reference plane](#).

LINE

The LINE and THRU standards establish the reference impedance for the measurement after the calibration is completed. TRL calibration is limited by the following restrictions of the LINE standard:

- Must be of the same impedance and propagation constant as the THRU standard.
- The electrical length need only be specified within 1/4 wavelength.
- Cannot be the same length as the THRU standard.
- A TRL cal with broad frequency coverage requires multiple LINE standards. For example, a span from 2 GHz to 26 GHz requires two line standards.
- Must be an appropriate electrical length for the frequency range: at each frequency, the phase difference between the THRU and the LINE should be greater than 20 degrees and less than 160 degrees. This means in practice that a single LINE standard is only usable over an 8:1 frequency range (Frequency Span / Start Frequency). Therefore, for broad frequency coverage, multiple lines are required.
- At low frequencies, the LINE standard can become too long for practical use. The optimal length of the LINE standard is 1/4 wavelength at the geometric mean of the frequency span (square root of $f_1 \times f_2$).

Note: The TRL LINE standard must have a delay that is greater than 0 (zero) ps. Otherwise, calibration correction calculations will contain unpredictable results.

MATCH

If the LINE standard of appropriate length or loss cannot be fabricated, a MATCH standard may be used instead of the LINE.

- The MATCH standard is a low-reflection termination connected to both Port 1 and Port 2.
- The MATCH standard may be defined as an infinite length transmission line OR as a 1-port low reflect

termination, such as a load.

- When defined as an infinite length transmission line, both test ports must be terminated by a MATCH standard at the same time. When defined as a 1-port load standard, the loads are measured separately. The loads are assumed to have the same characteristics.
- The impedance of the MATCH standard becomes the reference impedance for the measurement. For best results, use the same load on both ports. The load may be defined using the data-based definition, the arbitrary impedance definition, or the fixed load definition.

See Also

- See [Modify Calibration Kits](#) for detailed information about creating and modifying Calibration kit definitions.
- For more information, read [Specifying Calibration Standards and Kits for Agilent Vector Network Analyzers \(Application Note 1287-11\)](#)

TRL on a 4-port PNA-L and ALL PNA Models with an External Test Set

Beginning with the PNA code revision 5.25, TRL CAN be performed on a 4-port PNA-L and ALL PNA Models with an [External Test Set](#) enabled. Previously, a TRL calibration required a PNA with a reference receiver for each test port. With the new TRL method, a Delta Match Calibration is first performed and applied.

Note: See [Delta Match Calibration](#) to learn which models require this.

The accuracy of this TRL cal greatly depends on the accuracy of the Delta Match Calibration. With an accurate Delta Match Calibration, the difference in accuracy between a traditional TRL cal and this TRL cal is negligible.

How to Perform a TRL Cal in these cases

1. Click **Calibration, Cal Wizard**.
2. Select a TRL cal kit for the ports to be calibrated.
3. During the calibration, the Cal Wizard prompts you for a [valid Delta Match Cal](#).

Last modified:

13-Dec-2012	Fixed models for GDM cal.
3-Aug-2012	added line std note
16-Sep-2010	Slightly modified for cals on apps and added link to TRL Options.
14-Apr-2008	Added App Note link
9/12/06	with Ext Test Set

CalPod (Opt 301 or 302)

CalPod is a system that simplifies the process of recalibrating the PNA without requiring the removal of the DUT or the physical connection of standards. This allows recalibration from a remote location such as when the DUT is in a temperature chamber.

In this topic:

- [Overview](#)
- [How to start the CalPod dialog](#)
- [CalPod dialog](#)
- [CalPod Setup dialog](#)
- [CalPod Operational Check](#)

See Also

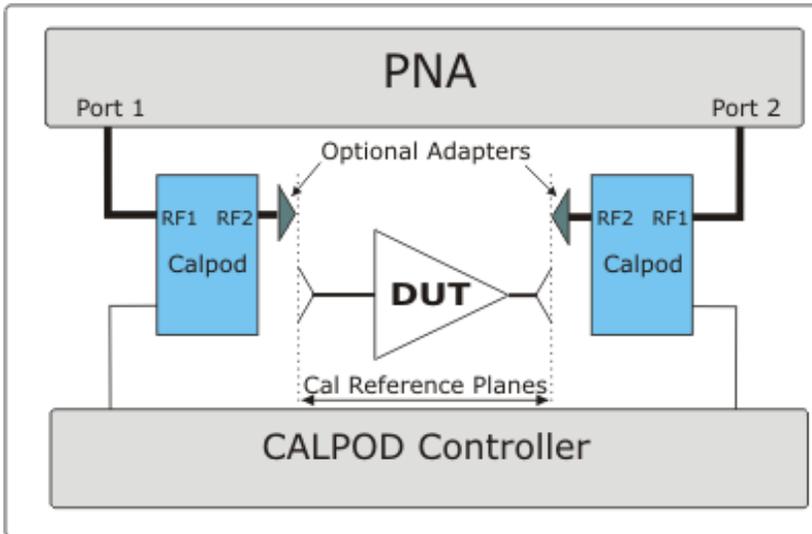
[CalPod as ECal](#)

Other Calibration topics

Process Overview

Note: The following overview assumes the CalPod system has been installed and configured. See the **CalPod User's Guide** for installation instructions at: <http://cp.literature.agilent.com/litweb/pdf/85523-90001.pdf>

The following process assumes a 2-port DUT connected to the PNA ports 1 and 2 through CalPod modules as follows:



The Blue boxes represent CalPod modules with internal Thru, Short, Open, and Load states.

1. After configuring and assigning CalPod modules to PNA ports 1 and 2, connect the CalPod modules to the PNA, directly or using short cables. [Learn how to configure CalPod.](#)
2. Setup measurements on a channel. An IFBW of 1 kHz or lower with eight averages is recommended. CalPod does not support measurements below 100 MHz.
3. Perform a full 2-port calibration for the channel with the CalPod outputs as the reference plane.
4. Click **Initialize Channel** to automatically perform the following steps:
 - a. The OPEN, SHORT, AND LOAD states of both Calpod modules are switched in and S11/S22 are measured.
 - b. The resulting measurements are stored in the channel's Cal Set as additional standard measurements. These measurements are used to characterize the Calpod states - they are NOT used at this time to change the error correction.

Notes:

- Because the OPEN, SHORT, AND LOAD states in the CalPods are measured, it is not important what is connected to the CalPod when Initialize is pressed. Therefore, for highest accuracy, click Initialize IMMEDIATELY and ONLY ONCE after performing the calibration - before causing ANY cable movement.
- If an adapter is required to connect the DUT to a CalPod, use a high-quality adapter. Any temperature drift due to the adapter is NOT recorrected.
- Always connect the DUT as close as possible to the CalPod modules.

5. Connect the DUT to the CalPod outputs.

6. Click **Recorrect Channel** or **Recorrect All Channels** whenever necessary. Any of the following actions will cause the current calibration to become invalid and require recorection:
 - a. Moving the CalPod modules to the ends of long cables.
 - b. Changing the cables.
 - c. Extreme temperature variations.
 - d. Measurement drift over long time periods.

7. The following steps occur automatically during recorection for the active channel:
 - a. The OPEN, SHORT, AND LOAD states of both CalPod modules are switched in and S11/S22 are measured.
 - b. Additional (de-embedded) error terms are computed to compensate for changed conditions from the Initialize measurements.
 - c. Another Cal Set is created using the original name with the CalPod number appended. The modified error terms are saved to that Cal Set and applied to the channel. The measurements are now fully corrected.

How to start the CalPod dialog

Using front-panel HARDKEY [softkey] buttons

1. Press CAL
2. then **[More]**
3. then **[CalPod]**
4. then **[CalPod...]**

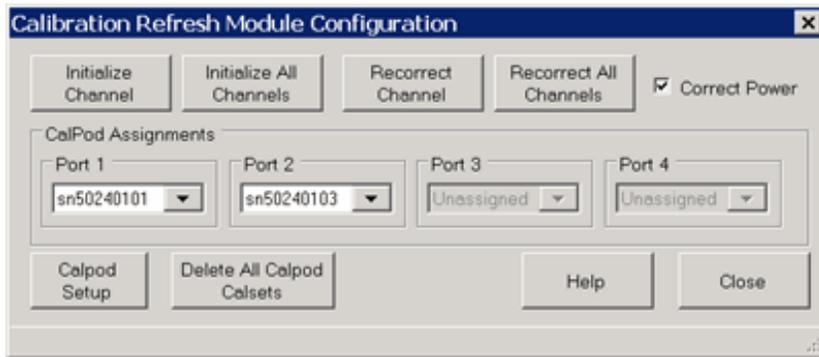
Using a mouse with PNA Menus

1. Click **Response**
2. then **Cal**
3. then **CalPod**
4. then **CalPod**

Programming Commands

CalPod dialog box help

[Learn all about the CalPod process.\(Scroll up\)](#)



Initialize Channel Calibrated measurements of the CalPod states are performed as initial reference data points for the active channel.

Initialize All Channels Calibrated measurements of the CalPod states are performed as initial reference data points for all current channels. This command is not recommended, it is generally preferable to initialize each channel immediately following calibration.

Recorrect Channel Recorrects the active channel Cal Set to match the initial reference.

Recorrect All Channels Recorrects the Cal Sets on ALL channels that were initialized.

Correct Power

This checkbox causes power to be recorrected **ONLY** when source power correction data is stored as error terms in the CalSet. This occurs only when a [Guided Power Cal](#) is performed and when an app channel is calibrated such as a FCA, GCA, IMD, and Noise Figure channel. This checkbox has **NO** effect when a S-parameter Cal or a standard [Source Power Cal](#) has been performed, because source power correction data is not stored in the CalSet.

When any of the above power calcs have been performed, and when this box is checked, the power output at the PNA port is adjusted to compensate for any change in path loss when Recorrect is performed. For example, if the path loss between the PNA port and the CalPod was increased by two dB following initialization, then the PNA output power will be increased by two dB upon recorection. Do this when you add a significant amount of loss in the calibration path, or when the power level at the DUT is important.

When a significant amount of loss is introduced in the calibration path, it may not be possible to increase the source power enough to overcome the loss. In this case, an **Unleveled source** message may appear on the PNA screen.

When the checkbox is cleared, the source power level is not corrected.

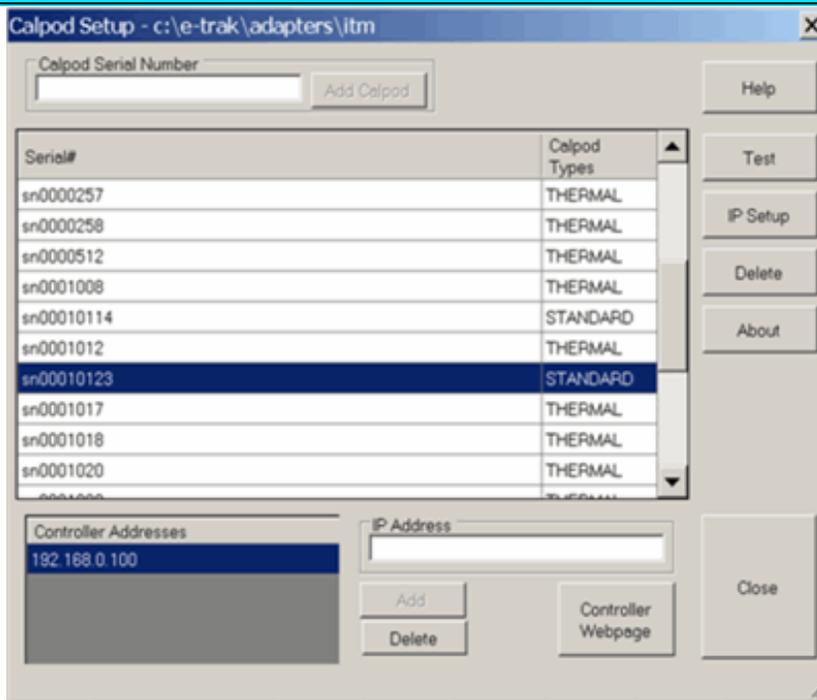
CalPod Assignments

For each PNA port, select a CalPod module.

CalPod Setup Starts the [CalPod Setup](#) dialog

Delete All CalPod Cal Sets Deletes all recorection Cal Sets and reinstates the Initialization Cal Set.

CalPod Setup dialog box help



To start this dialog, click **CalPod Setup** in the [CalPod](#) dialog box.

CalPod Serial Number Type the CalPod module (without 'sn'), then click **Add CalPod**. The new module is added to the list of available CalPod modules. (The list of available CalPod modules is limited to four entries when option 301 is the only PNA CalPod option.)

Serial # and CalPod Types

Shows the list of available CalPod modules. A CalPod module type may be STANDARD or THERMAL. A CalPod module will be listed as a STANDARD type unless two conditions are met: the PNA has option 302 and the thermal characterization data has previously been loaded into the PNA from a USB flash drive.

Buttons

Test Click to test the connection between the controller and the selected CalPod module. The

message box displays the connection status and temperature for both Ambient and Thermal modules. Only the Thermal module will apply test temperature for recorection.

Delete Removes the selected STANDARD CalPod module from the list.

To delete a THERMAL CalPod from the list:

1. Navigate to the c:/e-trak/adapters/itm directory.
2. Delete the .xml file associated with the CalPod serial number.
3. Exit all CalPod dialog boxes and restart the CalPod dialog.
4. The CalPod may now be removed using the **Delete** button.

About Shows the CalPod software version information.

For more CalPod Setup information, see the CalPod web site: <http://na.tm.agilent.com/pna/calpod>. Click **CalPod Controller Configuration**.

CalPod Operator's Check

This program is provided as a convenience to help determine the operational status of each 855xxA Series CalPod and its associated CalPod Controller. While this check is not intended to be a complete test, it does check each unit enough to provide greater than 95% confidence that the CalPod is functioning properly.

- When the max frequency of the CalPod is higher than the max frequency of the PNA, the full frequency range of the CalPod is not tested.
- Up to four CalPod modules may be checked at once. All four devices must be of the same frequency range.
- The software revision for the Operator's Check code is displayed in the upper left-hand corner of the window

Before running Op Check

The CalPod system must be installed and configured on the PNA. The PNA must have Option 301 or 302 installed.

See the **CalPod User's Guide** for instructions at: <http://cp.literature.agilent.com/litweb/pdf/85523-90001.pdf>

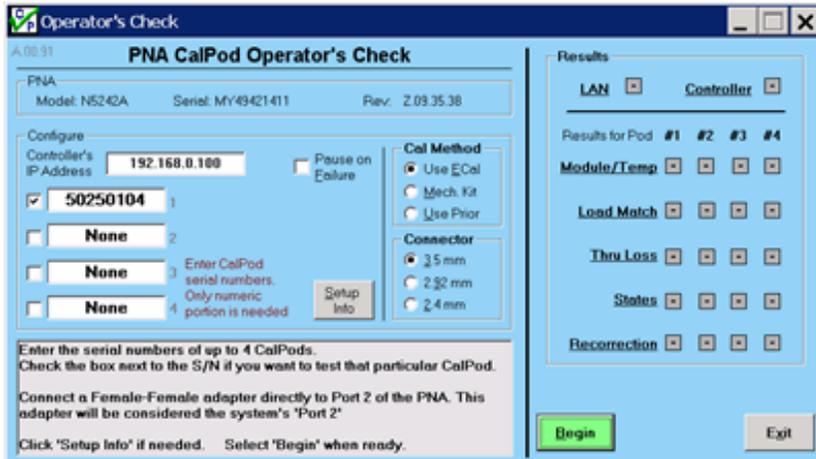
Required equipment:

- An appropriate ECal or mechanical Cal Kit.

- A high-quality cable.
- A female-female adapter of the calibration connector type.
- A fixed attenuator up to 10 dB (3 dB preferred) or other frequency insensitive device with similar loss.

How to perform CalPod Operators Check

Click **Utilities**, then **System**, then **Service**, then **CalPod**, then **OpCheck**.



Click **Setup Info** to learn more about this dialog.

Also, click **Cal Method** or **Connector** for additional explanation for these areas.

Configure

1. Enter information in the “Configure” area.
2. Each time a 2-port cal is performed, the results are saved in a file. The “Use Prior” selection uses the saved calibration.
3. When the calibration connector type does not mate with the CalPod connectors, perform the calibration and then use adapters to connect to the CalPod module.
4. Click **Begin** to start the Op Check.
5. Follow the prompts in the gray box.

Op Check Results

- The Results area shows Op Check progress.
- Click a test label for test information.
- When the check has finished, the results are saved to a text file. The default path and filename is:

C:/Program Files/Agilent/Network Analyzer/Service/calpodopchklog.txt. To save multiple results, rename the file or save it to a different location.

- For assistance in troubleshooting CalPod Operator's Check failures or for additional information, see the appropriate FAQ at the CalPod web site: <http://na.tm.agilent.com/pna/calpod>
-

Last Modified:

24-Apr-2013	Updated with SPC info (BH)
29-Mar-2013	Added link to Calpod as ECal
15-Jun-2011	Updated per LH
5-May-2011	New topic

CalPod as ECal (Option 304)

A [CalPod module](#) can be used as 1-port ECal module to calibrate channels for all measurement classes. In addition, the CalPod module can be left in place during measurements in order to refresh calibrations.

In this topic:

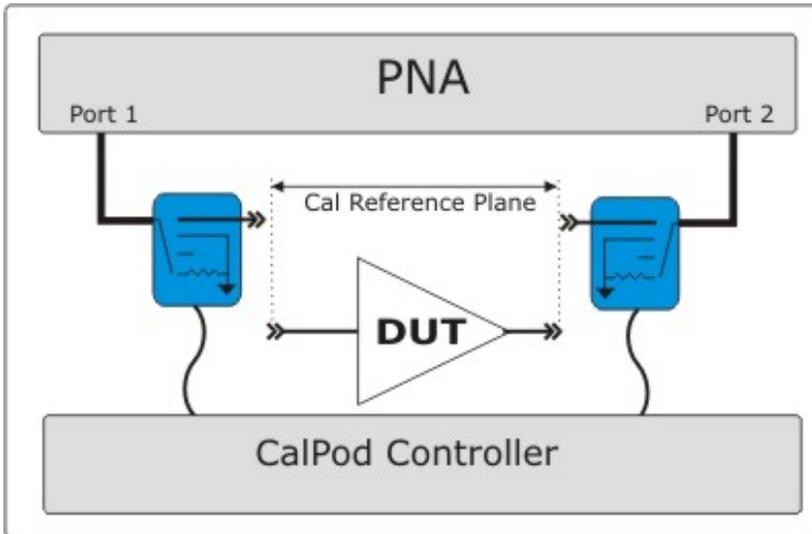
- [Overview](#)
- Characterizing the CalPod module
 - [Summary](#)
 - [Detailed Steps](#)
- [Calibrating with CalPod](#)

More Calibration topics

Overview

To use a CalPod module as a calibration device:

1. The CalPod module must be configured using the CalPod software. [Learn how](#).
2. Unlike an ECal module, a CalPod module has no internal memory and therefore, no internal characterization data. Before a CalPod can be used as an ECal module, its internal standards must first be characterized using the ECal User Characterization wizard. In addition, the characterization data can ONLY be stored in PNA disk memory.
3. After these steps have been performed, connect the CalPod modules to the controller to perform a PNA calibration. The CalPod modules will appear in Cal Wizard dialogs just like an ECal module.



This image shows TWO CalPod modules after both have been characterized. A PNA calibration, and subsequent 'refresh' calibrations can occur without making additional connections.

Characterizing the CalPod module - Summary

When performing a User Characterization for a CalPod module, you can establish the Calibration reference plane at the input connector, or at the output connector, depending on how you plan to use CalPod.

1. Initially calibrate the PNA.
2. After the calibration, the internal CalPod OPEN, SHORT, and LOAD standards are measured.
3. The measurements are saved to PNA disk memory. They are used when performing calibrations using the CalPod module.

Detailed steps to Perform a User Characterization

Note: If you have more than one CalPod module, each module must be characterized separately.

1. Perform a one-time CalPod configuration. [Learn how.](#)
2. Connect the CalPod module to the CalPod controller.
3. **Preset** the analyzer.
4. [Set up the measurement.](#) For best accuracy, the **IF bandwidth** should be set to **1 kHz** or less.
5. Start the User Characterization Wizard as follows:

**Using front-panel
HARDKEY [softkey] buttons**

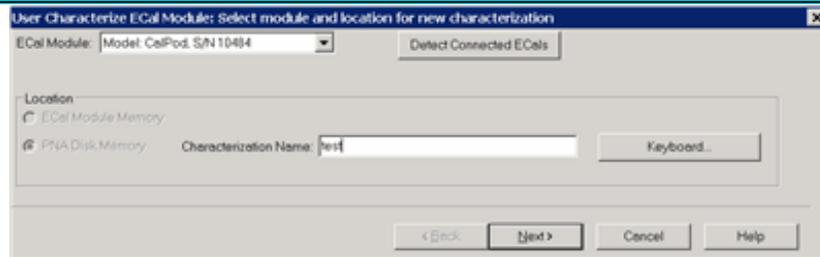
Using a mouse with PNA Menus

1. Press **CAL**
2. then **[More]**
3. then **[ECal]**
4. then **[Characterize ECal Module]**

1. Click **Response**
2. the **Cal**
3. then **More**
4. then **ECal**
5. then **Characterize ECal Module**

Programming Commands

Select Module and Location dialog box help



ECal Module Select a CalPod from list of connected modules and registered CalPods.

Detect Connected ECals Click to rescan CalPod/ECal modules.

Location

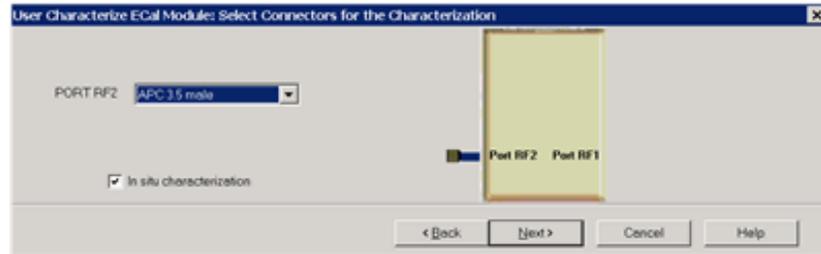
- **ECal Module Memory** NOT available for CalPod modules.
- **PNA Disk Memory** Enter a Characterization Name. This name appears when selecting a User Characterization to be used with subsequent calibrations.
 - [Learn how to manage characterizations that are stored to PNA disk memory.](#)
 - [See the benefits of storing the User Characterization to PNA Disk Memory.](#)

Keyboard Launches a keypad that can be used to type a characterization name from the PNA front panel.

Next Click to continue to the [Select Connectors for the Characterization](#) dialog box.

[See note regarding extended frequency use.](#)

Select Connectors for the Characterization dialog box help



In situ characterization

When Checked:

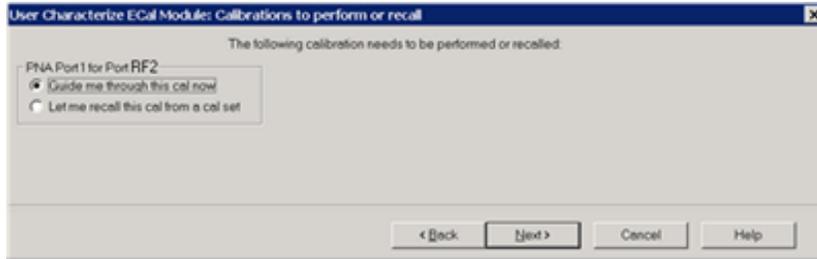
- This initial calibration, and all DUT measurements are performed WITH the CalPod module THRU path in place (In Situ).
- Select the RF2 (output) connector type and Cal Kit.
- The Cal Kit standards are connected to the RF2 Output connector, making it the reference plane. This calibration removes everything in front of the output connector from subsequent measurements.
- The User Char description will show "In Situ".

When Cleared:

- This initial calibration, and all DUT measurements are performed WITHOUT the CalPod module in place. This is similar to an ECal calibration.
- Select the RF1 (input) connector type and Cal Kit. The Cal Kit standards are connected at this location, making it the reference plane.

Next Click to continue to the [Calibrations to perform or recall](#) dialog box.

Calibrations to perform or recall dialog box help



Perform or load a 1-port cal.

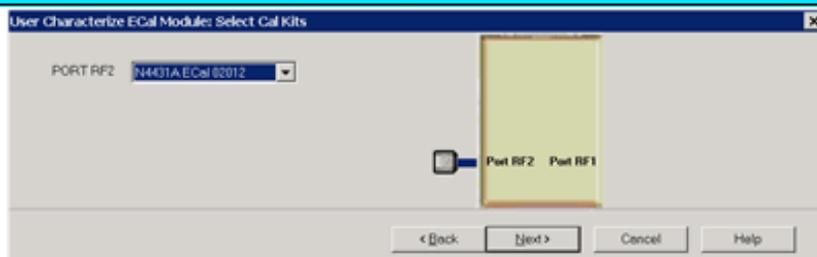
Guide me through this cal now Click to perform a Guided calibration. A calibration kit is required for each connector type.

Note: Some PNA-L models cannot perform TRL calibration during the calibration portion of a User Characterization. However, this type of Cal can be performed using the Cal Wizard, saved to a Cal Set, then recalled at this point in the User Characterization.

Let me recall this cal from a cal set Click to select an existing Cal Set. You cannot select a Cal Set that is currently in use. Learn more about [Using Cal Sets](#).

Next Click to continue to either the [Select Cal Kits](#) (Perform Cal) or [Select Cal Set](#) (Recall Calset).

Select Cal Kits dialog box help

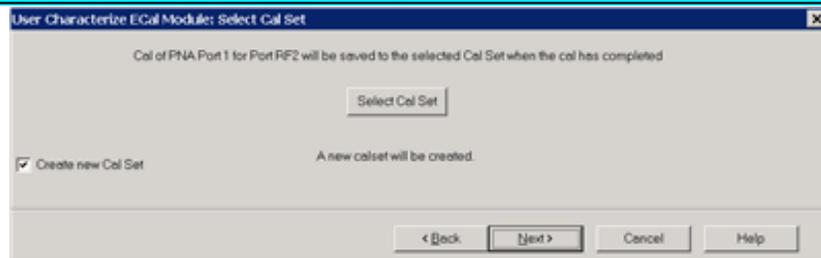


When 'In situ' is selected, a cal kit is selected for port "RF2", as this will be the calibration reference plane (where standards are connected).

Provides a list of calibration kits to perform the calibration. Select the Cal Kit you will use for the port.

Next Click to continue to the [Select Cal Set](#) dialog box.

Select Cal Set dialog box help



The calibration that you perform will be written to a Cal Set. This dialog box allows you to select a Cal Set to overwrite, or to write to a new Cal Set. The current choice is visible below the **Select Cal Set** button.

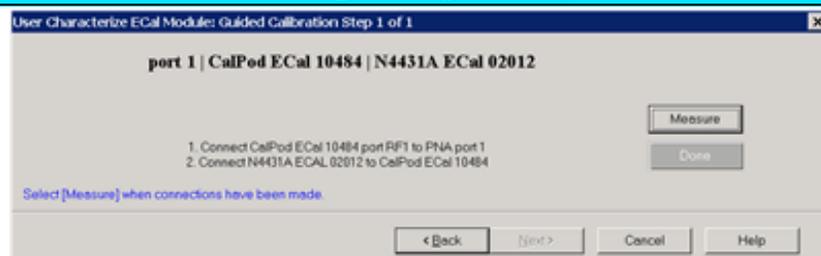
Select Cal Set Click to open the **Select A Cal Set** dialog box.

Create new Cal Set Check to create a new Cal Set to store the calibration. Clear to select and overwrite a stored Cal Set.

Next Click to continue to the [Guided Calibration Steps](#) dialog box.

Note: Remember the Cal Set name for future reference.

Guided Calibration Steps dialog box help



The first Measure page.

Connect each calibration standard to the location in the prompt.

Measure Click to measure the standard.

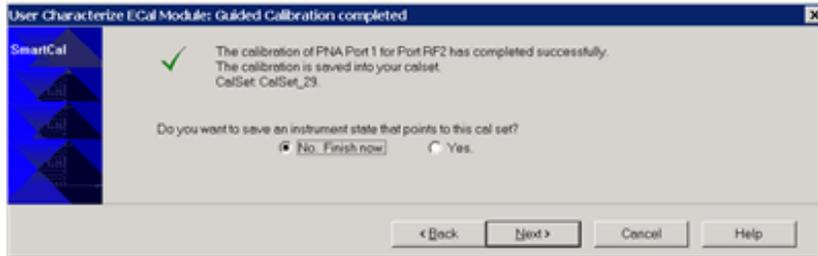
Back Click to repeat one or more calibration steps.

Done Click **after** a standard is re-measured and all measurements for the calibration are complete.

Next Click to continue to the next calibration step. (Does **not** measure the standard.)

Cancel Exits Calibration Wizard.

Guided Calibration completed dialog box help



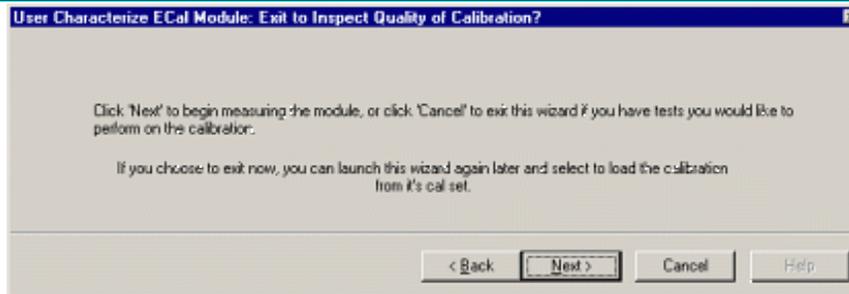
Allows you to finish the calibration and continue to the next characterization steps.

No. Finish now Select to save Cal Set data.

Yes Allows selection of Save options.

Next Click to continue to the [Exit to Inspect Quality of Calibration](#) dialog box.

Exit to Inspect Quality of Calibration dialog box help



Allows you to exit User Characterization to [validate the calibration](#) before proceeding with the characterization.

Back Allows you to repeat calibration.

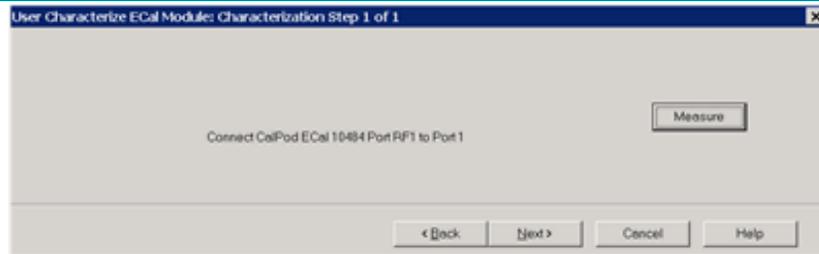
Next Click to continue to the [Characterization Steps](#) dialog box.

Cancel Exits the Calibration.

To return to the current step:

1. Start User Characterization.
2. In the **Select Connectors for Characterization** dialog box, click **Next**. (Previous entry is stored in memory.)
3. In the **Calibrations to perform or recall** dialog box, recall the Cal Set that you just performed.

Characterization Steps dialog box help



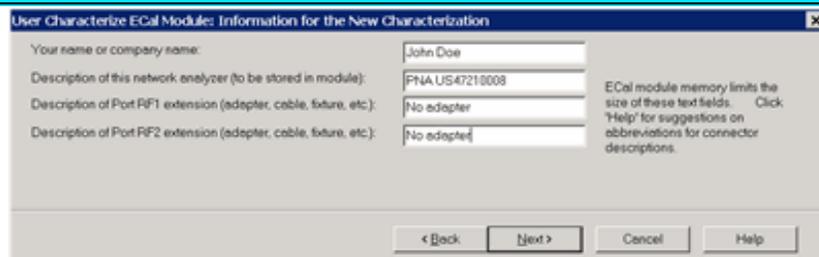
After the initial PNA calibration, the CalPod internal states are measured.

Connect the CalPod if not already connected.

Measure Measures the CalPod module.

Next Click to continue to the [Information for the New Characterization](#) dialog box when measurements are complete.

Information for the New Characterization dialog box help



Enter descriptive information that will be stored in the characterization file. This description will be viewable in the several PNA dialogs when the CalPod is selected.

Next Click to continue to the [Write Characterized Data](#) dialog box.

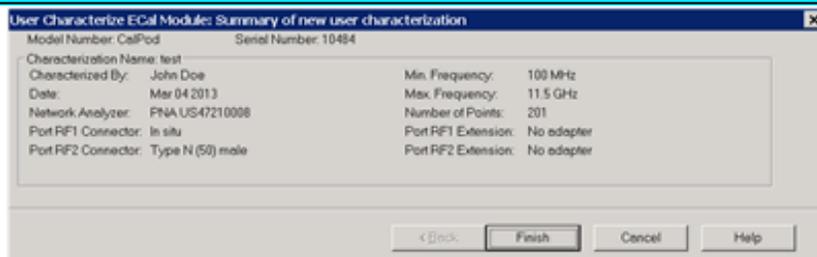
Write Characterized Data dialog box help



The PNA writes User Characterization data to the PNA disk memory.

Write Click to write data.

Summary of new User Characterization dialog box help



Summary page of characterization wizard after characterization is complete.

Cancel Click to exit (characterization complete).

Finish Click to exit (characterization complete).

Calibrating with CalPod

The CalPod module is now characterized and ready to be used in a PNA calibration.

When CalPod modules are selected for ALL ports of an N-port cal, Unknown Thru is the only selectable Thru method for all paths. [Learn more about Unknown Thru.](#)



The above image is the **Connectors and Cal Kits** page as seen during a SmartCal on a standard channel. A similar version of this page is shown in the Calibration Wizard for all Application channels (FCA, GCA, NFA and so forth).

Last Modified:

12-Feb-2013 New topic

Calibration Preferences Wizard

Two Cal Preferences are set from this Wizard:

1. [Whether or not to show the first 'Method' Page of the Cal Wizard](#)
2. [Select and order the Cal Types that are available during a SmartCal with Mechanical Standards](#)

To change either of these choices, you must select **Yes, Enable the calibration preferences** at the first Wizard page.

How to change Cal Preferences

Programming commands are NOT available for the preference settings discussed in this topic, although there are other [Cal Preferences](#) that can be set remotely.

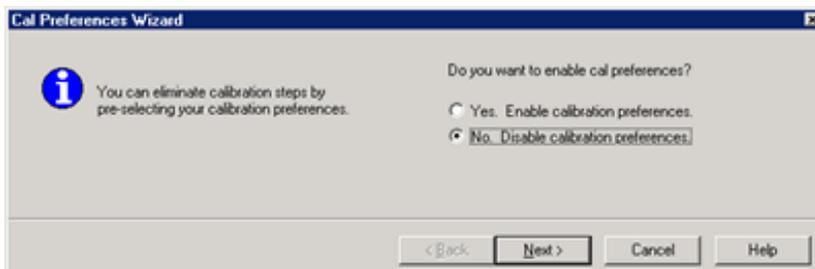
Using front-panel HARDKEY [softkey] buttons

1. Press **CAL**
2. then **[Start Cal]**
3. then **[Preferences]**

Using a mouse with PNA Menus

1. Click **Response**
2. then **Cal**
3. then **Start Cal**
4. then **Preferences**

◀ **Programming Commands** ▶

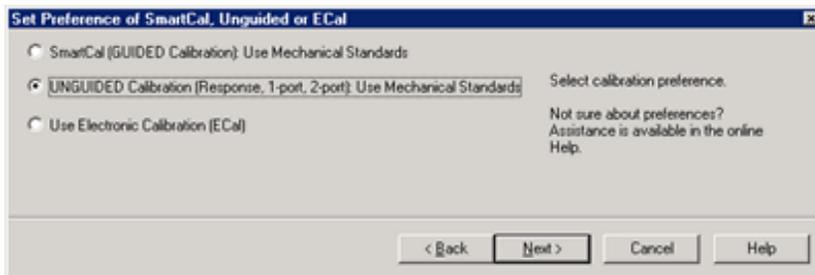


Cal Preferences Wizard dialog box help

Use this dialog to change either of the following preferences:

- Show or Hide the first page of the Cal Wizard
- Select order of calibrations that are offered.

To change either of these choices, you must select **Yes, Enable the calibration preferences**.



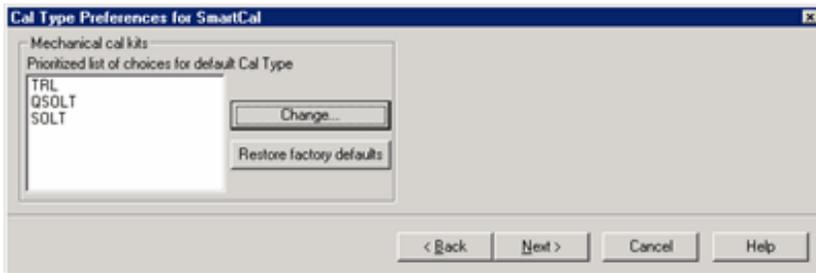
Cal Preferences of... dialog box help

Use this dialog to change which Cal method to perform.

After making this selection, the first page of the Cal Wizard will not be shown on subsequent calibrations.

To change **ONLY** the order of Cal Types that are offered, and none of these preferences, you must do the following:

1. Select one of these choices and click **Next**.
2. Select and order the Cal Types, then click **Next**
3. Click **Finish**
4. Click **Cal**, then **Cal Wizard**.
5. On the first Cal Wizard page that shows, click **Back**, then clear the **Preferences** checkbox.



Cal Type Preferences dialog box help

This dialog is used to set which Cal Types are available, and the order in which they are selected as the default choice, during a SmartCal with Mechanical Standards. This setting is also used to set the default Cal Type for Guided calibrations using SCPI or COM.

The specified Cal Type order should allow you to make fewer changes to the Cal Type during a SmartCal with Mechanical Standards.

For example, in the above image, the first Cal Type on the list is TRL. When doing a SmartCal with Mechanical Standards:

- If a TRL Cal Kit is available for the specified DUT connectors, then TRL becomes the default Cal Type.
- If a TRL Cal Kit is NOT available, then the second Cal Type on the list (SOLT) is evaluated for compatibility with the available Cal Kits, and so forth with the Cal Types that remain on the list.
- If TRL is removed from the list, that Cal Type is NOT available for selection during a SmartCal with Mechanical Standards.

[Learn more about Cal Types.](#)

[See where you choose Cal Type during a SmartCal](#)

Prioritized list of choices for default Cal Type Shows the current list of Cal Types and the order in which they will be selected for Mechanical calibrations.

Change Click to invoke the [Modify list of default Cal Types](#) dialog.

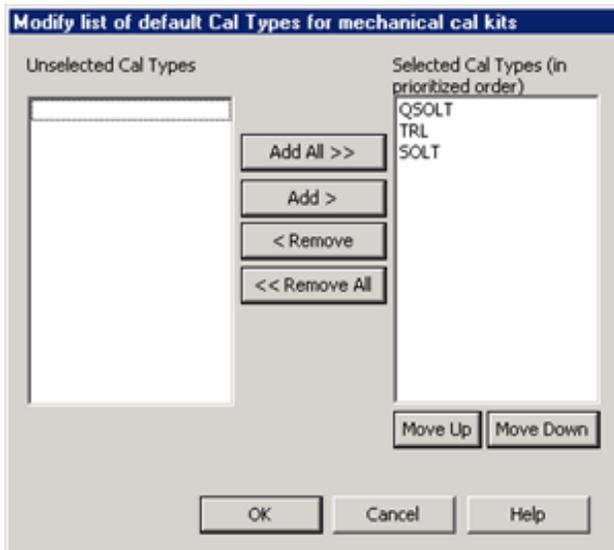
Restore factory defaults Returns the list to the original selections and order. The factory defaults are in order of accuracy from highest (TRL) to lowest (QSOLT).

Cancel Closes the dialog without making changes.

Notes:

- Your Cal Type settings are saved only until the PNA application is closed. When re-opened, the factory default settings are restored.

[Learn more about QSOLT Calibration](#)



Modify list of default Cal Types dialog box help

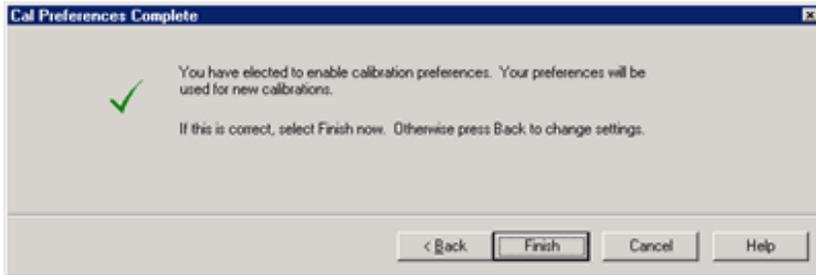
Use this dialog to Add, Remove, and re-order the available Cal Types. There must be at least ONE selected Cal Type to perform a SmartCal with Mechanical Standards.

Unselected Cal Types Cal Types in this list will not be presented as a choice during a Calibration.

Selected Cal Types Cal Types in this list will be presented, in order, as the default choice during a Calibration. Click a Cal Type to select it, then click the following buttons to perform that operation.

Add / Remove buttons Click to Add and Remove the selected Cal Types from the Selected Cal Types list.

Move Up / Down Click to re-order the Selected Cal Types list.



Cal Preferences Complete dialog box help

Either Enable or Disable Cal Preferences.

See how to [select ONLY Cal Type Preferences](#) and continue to show the first Cal Wizard page.

Last Modified:

3-Sep-2008	Removed legacy content
19-Mar-2008	Small cosmetic edits
1-Jan-2007	MX added UI

Power Calibration

Source and Receiver Power Calibrations work together to provide very accurate power levels from the source, and very accurate power measurements from the PNA receivers.

- [Source Power Calibration Overview](#)
- [Supported Power Meters and Sensors](#)
- [How to perform Source Power Calibration](#)
- [Setup](#)
- [Source Power Cal dialog](#)
 - [Source Power Calibration Options dialog](#)
 - [Power Meter Settings dialog](#)
 - [Power Loss Compensation dialog](#)
 - [Power Sensor Settings dialog](#) (Zero / Calibrate)
- [Copy a Source Power Calibration to other Channels](#)
- [Saving a Source Power Calibration](#)
- [Reducing Time to Complete a Source Power Calibration](#)
- [Receiver Power Calibration](#)
- [Saving Receiver Cals](#)

Other Source Power Cal choices

- **Guided Power Cal** can be performed during an S-parameter Guided Calibration. [Learn more.](#)
- **Receiver Leveling** can be used to provide 'real-time' source power cal. [Learn more.](#)
- **See Also:** [Configure an Power Meter As a Receiver \(PMAR\)](#)

[See other Calibration Topics](#)

Source Power Calibration Overview

Perform Source Power Calibration when you need accurate power levels at some point in the

measurement path between the PNA test ports. For example, you need to characterize the gain of an amplifier across a frequency range at a specified input power. You would perform a source power cal at the input of the amplifier to ensure the **exact** power level into the amplifier across the frequency range.

Using a Source Power Cal, you can expect the power at the point of calibration to be within the range of the uncertainty of the power meter and sensor that is used.

Source Power Calibration...

- Is independent of measurement type. It corrects the PNA source regardless of which receivers are being used in a measurement. Therefore, it can be used with both [ratio or non-ratio measurements](#).
- Applies ONLY to those measurements on the selected channel that use the test port that was [specified as the Source](#) for the calibration. For example, if you specify Channel 1 and Port 1 as the source to be calibrated, only those measurements on channel 1 that use port 1 as the source will be corrected.
- Can be used in conjunction with other measurement calibrations, such as a full 2-port calibration. For highest accuracy, perform the measurement calibration AFTER the source calibration.
- Can be used with [Power Sweep](#) type. Source Power Cal will correct the power at all power levels across the power sweep.
- Can be used with [Port Power Uncoupled](#).
- Forces [sweep mode to Stepped](#) on measurements with source power correction turned ON.
- Beginning with PNA Rev. 7.50, an external source can be calibrated using Source Power Cal.

Overview of How it works:

[See Important First-time USB connection note.](#)

[Click to see the detailed procedure](#)

1. Specify the measurement settings (frequency range, IFBW and so forth).
2. Start Source Power Calibration.

Note: When using an Agilent 848X power sensor (sensors that do NOT have built-in calibration factors), enter the Cal Factors using the [Power Sensor Settings](#) dialog, because the PNA instructs the power meter to NOT use the Cal Factor tables internal to the power meter.

3. Connect a power meter sensor to the point at which you want a known power level. This may be at the input or output of your device, or some other point between the test ports.
4. The PNA source is stepped through the specified frequency range, and power is measured with the power meter. At each data point, the source power is adjusted until the measured power is within your specified accuracy level.

5. When complete, the power meter is preset. The source power calibration can be [saved as part of the instrument state](#).
6. The power meter is removed and the measurement path reconnected.
7. The calibration is automatically applied to the channel. All measurements on that channel using that source port benefit from the source power cal.
8. Perform an S-parameter calibration AFTER a Source Power Cal. The S-parameter cal is performed using the corrected stimulus power levels for the relevant ports.

Verify the source power calibration using the following procedure.

1. Connect the power meter as it was during the source power calibration.
2. Set the PNA to [Point Trigger](#) mode.
3. Trigger the PNA across the trace. Read about the behavior of the [sweep indicator](#).
4. At each data point, the power meter should read the corrected power level within the specified tolerance.

Supported Power Meters and Sensors

[See Agilent's Power Meters and Sensors Webpage.](#)

USB Power Sensors

- U848x Series USB Thermocouple Power Sensors (A.09.90.08 and later).
 - These include the following models: U8481A, U8485A, U8487A, U8488A
 - External Calibration (connecting the sensor to the 1 mW ref port) is NOT supported.
- U2020 X-Series USB Peak and Average Power Sensors.
 - The PNA does NOT support peak mode in these sensors, but measures average power.
- U2000 Series USB power sensors.

USB Notes:

- From a standard power cal (this topic), only one USB power sensor can be used to cover the entire frequency span. To use multiple power sensors, perform a Guided Power Cal. [Learn how.](#)
- To select a USB power sensor for a standard power cal:

1. Connect the sensor directly to one of the PNA USB ports.
2. From the [Source Power Cal](#) dialog, click **Power Meter Config**.
3. On the [Power Meter Settings](#) dialog, select **USB**.

- [See Important First-time USB connection note](#).
- See note about [Zeroing USB Power Sensors](#).
- See also: [Power Meters as Receivers](#) (PMAR)

Power Meters

- P Series power meters (N1911A and N1912A) and all supported sensors.
- EPM Series power meters (N1913A and N1914A) and all supported sensors.
- EPM-P Series power meters (E4416A and E4417A) and all supported sensors.
- E Series power meters (E4418 and E4419) and all supported sensors.
- HP 437B / 438A power meters.

Power Meter Notes:

- N1911A, 12A, 13A, and 14A power meters have a **device-side USB connector**  and are controlled by the PNA exactly like a USB sensor. See [USB Power Sensors](#) (above). Although these meters may also have a front-panel USB port, USB power sensors must be connected directly to one of the PNA USB ports.
- Source Power Calibration operates slowly with the Agilent E930x and E932x power sensors.
- Some Agilent power meters have a mode that emulates the command set of the 437B or 438A power meter. The PNA does NOT support this emulation mode.
- The [82357A USB/GPIB Interface](#) can be used to control power meters.
- [Create a Custom Power Meter Driver](#) for use with other power meters.

Non-Agilent Power Sensors

- Rohde and Schwarz NRP-Z power sensors (limited support). [Learn how to install the drivers](#).

How to perform Source Power Calibration

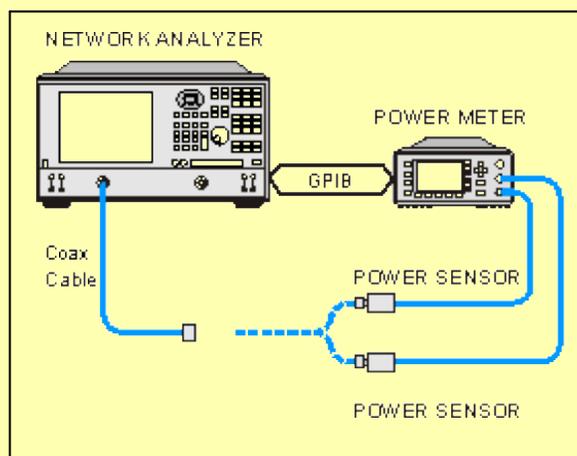
Note: Beginning with PNA firmware rev A.09.30, an Guided Power Cal can be performed during

an S-parameter Guided Calibration. [Learn more.](#)

1. Setup your measurement (sweep type, frequency range, IFBW, and so forth). By default, a Source Power Cal is performed on the source port of the active measurement.
2. Connect coax cable, GPIB cable, and power sensors to the PNA as shown in graphic below.

This image does NOT apply to USB power sensors, which are connected directly to a PNA USB port.

[See Important First-time USB connection note.](#)



3. Apply power to the power meter and allow 30 minutes warm-up time before beginning calibration.
4. Select **Source Power Cal** as follows:

**Using front-panel
HARDKEY [softkey] buttons**

1. Press **CAL**
2. then **[Power Cal]**
3. then **[Source Cal]**

Using a mouse with PNA Menus

1. Click **Response**
2. then **Cal**
3. then **Power Cal**
4. then **Source Cal**

Programming Commands

5. Complete the Source Power Cal dialog box (below), including [Options](#), [Loss Compensation](#) and [Power Sensor Settings](#), as needed.

Note: When using an Agilent 848X power sensor (sensors that do NOT have built-in calibration factors), enter the Cal Factors using the [Power Sensor Settings](#) dialog, because the PNA instructs the power meter to NOT use the Cal Factor tables internal to the power meter.

6. When complete, click **Take a Cal Sweep** in the Source Power Cal dialog box.
7. Follow the prompts to connect the sensors as required.
8. At this time you can change the Source Port setting and perform a Source Power Cal on a different port.
9. When calibration is finished, click **OK**. Correction is then applied and turned ON for the relevant ports on the active channel.
10. Remove sensor.
11. **SrcPwrCal** is displayed in the status bar when Source Power Correction is applied to the Active Measurement.
12. Perform a S-parameter calibration, which would use the corrected stimulus power levels for the relevant ports.

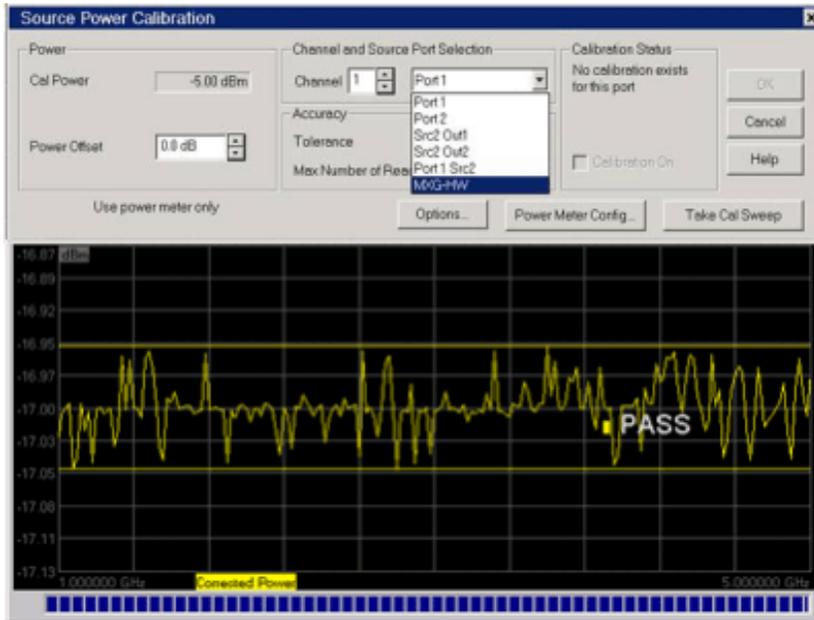
To turn Source Power Correction OFF:

- On the **Calibration** menu, point to **Power Calibration**, then click **Source Power Correction on/OFF**.
- ONLY correction for the source port of the ACTIVE MEASUREMENT is turned OFF (regardless of [port power coupling setting](#).)

Interpolation or Extrapolation

If the original stimulus settings are changed, Interpolation or EXTRAPOLATION is applied and **SrcPwrCal*** is displayed in the status bar. This is different from [measurement calibration interpolation](#). For example, if the frequency span is increased, the PNA will extrapolate new correction values rather than turn correction off. This is to protect your test device from being overpowered by the source. If the original settings are restored, then source power calibration returns to full correction.

Source Power Cal dialog box help



Note: Be sure that the frequency range of your power sensor covers the frequency range of your measurement. This does NOT occur automatically.

Power

Cal Power The calculated power (in dBm) at the calibration point. This value is the specified PNA source power plus the Power Offset value.

Power Offset Allows you to specify a gain or loss (in dB) to account for components you connect between the source and the reference plane of your measurement. These components will remain during a measurement. For example, specify 10 dB to account for a 10 dB amplifier in the path to your DUT. Following the calibration, the PNA power readouts are adjusted to this value.

To account for components that will be removed when the calibration is complete, use the [Loss Compensation table](#).

Channel and Port Selection

Channel Specifies the channel on which to perform the calibration. This setting defaults to the active channel.

Source Port Specifies the source port to be corrected. This setting defaults to the source port for the active measurement.

Beginning with PNA Rev. 7.22, external sources can be calibrated using this dialog. [Learn more](#).

Accuracy

At each data point, power is measured using the [specified Power Meter Settling Tolerance](#), then

adjusted until the reading is within this Accuracy **Tolerance** or the **Max Number of Readings** has been met. The **last** power reading is plotted on the screen against the Tolerance limit lines.

Tolerance Sets the maximum desired deviation from the specified **Cal Power** level in 0.005 dB increments from 0 to 5 dB.

Max Number of Readings Sets the maximum number of readings to take at each data point for iterating the source power. Enter a value between 1 and 1000.

Calibration Status

Allows you to turn Source Power Cal ON | OFF and view Cal data for each port, regardless of the active measurement. This feature allows the [Internal Second Source](#) to be calibrated and turned ON | OFF, even when being used as an incidental source in a measurement, such as an LO.

Calibration ON Check to turn Source Power Calibration ON for the specified source port.

The displayed text indicates when [interpolation](#) is applied for the calibration.

Buttons

Options Invokes the [Source Power Cal Options](#) dialog. Label to the left of the button displays the current 'Options' setting.

Power Meter Config Invokes the [Power Meter Settings](#) dialog box

Take Cal Sweep Begins source power calibration measurement.

OK Applies calibration. This button is disabled until the Take Cal Sweep has been pressed.

Cancel If a sweep is in progress, cancels the sweep. Press again to close the dialog.

Attention please: the power meter is operating in 200 r/s mode.

During a measurement, some Agilent power meters may display this message on the screen: It means that the meter is operating in 200 readings/sec which is the fastest speed setting for this meter. This is normal operation.

Pass / Fail Limits

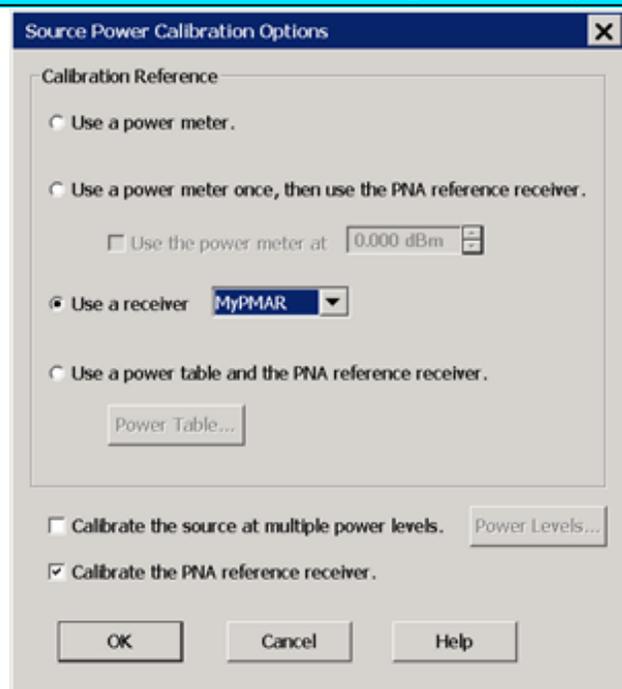
Limit lines are drawn on the Source Power Cal measurement graticule area. These lines are at the Cal Power +/- the current setting of Accuracy Tolerance. A FAIL during the Source Power Cal sweep means that the PNA was unable to measure power to within the Accuracy Tolerance. Tight tolerances are more difficult to achieve at lower Cal Power levels. When a FAIL indication appears,

increase the Max Number of Readings. If this does not cause a PASS condition, then decrease the Accuracy Tolerance value.

See Also

- Learn more about [Source Power Cal](#)
- Learn about [External Testsets](#) and Source Power Cal.

Source Power Calibration Options dialog box help



Provides options for measurement of the source power.

Note: At low power levels (less than -30 dBm) most power meters are not as accurate as a PNA receiver.

Calibration Reference Choose power meter/PNA receiver to use to measure power.

- **Use a power meter.** Traditional source power calibration using only a power meter to measure the source power at each data point. Most accurate (at higher power levels) and slowest method.

Note: Because the following two settings use PNA receivers to make power measurements, they do NOT work correctly when a [Frequency Offset](#) value is being used.

- **Use a power meter once, then use the PNA reference receiver.** When checked, the first reading at each data point uses a power meter to calibrate the reference receiver. Subsequent readings, if necessary to meet your accuracy requirement, are measured using the reference receiver. This technique is much faster than using the power meter, and more accurate when measuring low power levels.

Note: Do NOT use this setting if there is a component before the power sensor that exhibits non-linear behavior, such as a power amplifier in compression. Use a power meter and [Calibrate the source at multiple power levels](#).

- **Use a receiver.** Select a PNA Receiver or a PMAR (Power Meter as Receiver).

PNA receiver - For highest accuracy, first calibrate the receiver by performing a source power cal using a power meter, then a [receiver cal](#). That receiver can then be used to quickly calibrate other PNA source ports, or used on another channel with different stimulus settings. This would be useful, for example, if the power level of the measurement was below the sensitivity of the power sensor. Calibrate the PNA receiver using a source power cal that is within the sensitivity of the sensor. Then, use the calibrated receiver to perform a second source power cal at the reduced power level.

- The PNA receiver is specified using either standard receiver notation or [logical receiver notation](#).
- It is best to use the reference receiver for the source port to be calibrated. For example, if calibrating source port 2, specify "R2" or "a2" which is the same port 2 reference receiver using [logical receiver notation](#).
- To ensure an accurate source power cal, the frequency range over which the receiver was calibrated must be the same or larger than the "receiver only" source power calibration.
- All accuracy and settling tolerance and number of reading settings apply just as they do with a power meter reading.

PMAR Device - The power meter/sensor must first be configured. [Learn how to Configure a PMAR device](#).

- **Use a power table and the PNA reference receiver** Used to provide power leveling with mmWave test set and modules. [Learn more](#).

Calibrate the source at multiple power levels Used primarily with [mmWave measurements](#).

This feature can also be used with standard PNA measurements when a component is used in the source path such as a booster amp which does NOT have linear gain or loss over frequency. If this is not true for your setup but want to improve your source power accuracy, consider using the [Receiver Leveling](#) feature.

When checked, source power is measured using the specified 'Cal Reference' device (power

meter/sensor or PNA receiver) and iterated on a sweep-to-sweep basis to construct a 2-dimensional power table: Power IN, Power OUT, over all frequencies.

- Click **Power Levels** to launch the [Source Cal Power Levels dialog box](#) to set the power levels at which source power is to be measured.
- The source power cal is saved, but the power table is NOT accessible.

Calibrate the PNA reference receiver Check to calibrate the appropriate reference receiver to the power level that is measured at the calibration plane. Do this to make very accurate measurements using the calibrated reference receiver. This cal is done in addition to the standard source power cal using the any of the methods listed above. At the end of the source power cal measurement sweep, you can optionally save the reference receiver cal to a Cal Set to be recalled at a later time. The Cal is saved when the **OK** button is clicked to close the Source Power Cal dialog.

Source Cal Power Levels dialog box help



This dialog appears when you click **Power Levels** on the [Source Power Cal Options dialog](#).

Specify the power levels at which the Source Power will be calibrated. These values should be set to a few dB more or less than the measurement power levels.

Max Power - The highest power level at which to calibrate. This value should be a few dB **higher** than the highest power level of your measurement.

Min Power - The lowest power level at which to calibrate. This value should be a few dB **lower** than the lowest power level of your measurement.

Power Step - Calibrate at every incremental power level, between the Max and Min Power settings.

Power Meter Settings dialog box help

The screenshot shows the 'Power Meter Settings' dialog box. It features a title bar with a close button. The 'Interface' section includes radio buttons for 'GPIB', 'USB', 'LAN', and 'Any'. The 'GPIB' option is selected, and its 'Address' is set to 13. The 'USB' option has a dropdown menu. The 'LAN' option has a text field for 'Hostname or IP Address'. The 'Any' option has a text field for 'VISA resource string/alias'. A 'Sensors' button is located below the interface options. The 'Settling' section contains a 'Tolerance' field set to 0.050 dBm and a 'Max Number of Readings' field set to 3. The 'Sensor Loss Compensation' section has a checkbox for 'Use Loss Table' which is unchecked, and an 'Edit Table' button. At the bottom of the dialog are 'OK', 'Cancel', and 'Help' buttons.

This dialog appears when you click the **Power Meter Config** button on many dialog boxes.

Communication

- **GPIB / Address** Select GPIB power meter. Then select the address for the power meter. Default is 13. The PNA will search VISA interfaces that are configured in the Agilent IO Libraries on the PNA. **Note:** Use this selection when using a [82357A USB/GPIB Interface](#),
- **USB** PNA scans for USB power sensors or [N191x device-side USB power meters](#). Select a power sensor from the list. Only ONE USB power sensor can be configured to cover the entire frequency range of the calibration. To use multiple power sensors, perform a [Guided Power Cal](#).
- **LAN** Specify the Hostname or IP address of the Power Meter.
- **Any** For future use.

Sensors Invokes the [power sensor settings](#) dialog box.

Settling

These Settling settings do not apply when a PNA receiver is the power measurement device. Each power meter reading is "settled" when either:

- two consecutive meter readings are within this **Tolerance** value or

- when the **Max Number of Readings** has been met.

The readings that were taken are averaged together to become the "settled" reading. The settled reading is then compared to the [Accuracy Tolerance requirements](#) (tolerance and max readings) specified on the Source Power Cal dialog box.

Tolerance When consecutive power meter readings are within this value of each other, then the reading is considered settled.

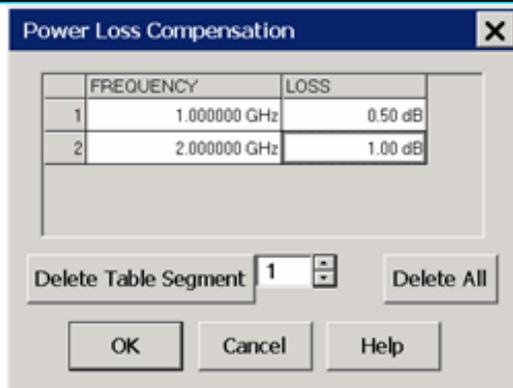
Max Number of Readings Sets the maximum number of readings the power meter will take to achieve settling.

Sensor Loss Compensation

Use Loss Table Select this checkbox to apply loss data to Source Power calibration correction (such as for an adapter on the power sensor).

Edit Table Invokes the [Power Loss Compensation](#) dialog box.

Power Loss Compensation dialog box help



To Add a Row to the table, click on a row in the table and press the down arrow on either the PNA front panel or keyboard.

To Edit a value, double-click in the cell to be edited.

Compensates for losses that occur when using an adapter or coupler to connect the power sensor to the measurement port. These components will be removed when the calibration is complete. To account for components that will remain during the measurement, use the [Power Offset setting](#).

The Frequency / Loss pairs define the amount of loss for the entire frequency range. For example, using the entries in the above dialog image:

- 0.5 dB is used to compensate power sensor measurements up to 1 GHz.
- Each data point between 1 GHz to 2 GHz is linearly interpolated between 0.5 dB and 1 dB.
- 1 dB is used above 2 GHz.
- A single frequency/loss segment is applied to the entire frequency range.

Beginning with A.09.80, enter up to **9999** segments to achieve greater accuracy. Previously the limit was 100.

These values can be loaded from an S2P file using the [Characterize Adaptor Macro](#).

Note: Large segment counts with one or more power sensors can result in long load and close times for the PNA Application.

Frequency Enter a frequency in Hz.

Loss Enter a loss as a POSITIVE value in dB. To compensate for gain, use NEGATIVE values.

Delete Table Segment Deletes row indicated in the field.

Delete All Deletes all data in the table.

The Power Loss Compensation table survives PNA Preset and Power OFF. To NOT use Loss compensation, clear the Use Loss table checkbox on the [Power Meter Settings](#) dialog.

Power Sensor Settings dialog box help

Sensor A
Cal factors are contained within this sensor

Use this sensor only (no frequency checking)

Minimum Frequency

Maximum Frequency

Internal Zero External Zero

Sensor B
Reference Cal Factor

	FREQUENCY	CAL FACTOR
1	50 000 000 Hz	113.3 %
2	51 000 000 Hz	108.9 %
3	52 000 000 Hz	109.1 %
4	53 000 000 Hz	106.5 %

Delete Cal Factor

Use this sensor only (no frequency checking)

Minimum Frequency

Maximum Frequency

Internal Zero External Zero

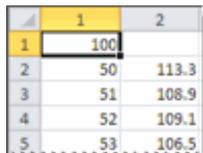
This dialog appears when you click the **Sensors** button on the [Power Meter Settings dialog](#).

Note: Be sure that the frequency range of your power sensor covers the frequency range of your measurement. This does NOT occur automatically.

Sensor A (B) Displays one of the following messages depending on type of sensor.

- **Not connected** The PNA is not detecting a power sensor.
- **Cal factors are contained within this sensor** This message is displayed when the Internal Reference Cal Factor and Cal Factor data are contained in the sensor and automatically accessed.
- **Sensor Data** Allows the following entries for power sensor data:
 - **Reference Cal Factor** Specifies the sensor's Reference Cal Factor.
 - **Cal Factor Table** Specifies the frequency and corresponding Cal Factor for the sensor.
 - **Delete Cal Factor** Deletes the indicated row in the table.
 - **Delete All** Deletes all data in the table.
 - **To Add a Row** to the table, click on a row in the table and press the down arrow on either the PNA front panel or keyboard. A row is added to the bottom of the table. The table is automatically sorted by frequency when OK is pressed.

Load Cal Factors Click to load cal factors from a *.csv file that you create from the cal factors that appear on the sensor. The first line of the file **MUST** have the reference Cal Factor (typically 100), followed by Freq / Cal Factor pairs as show in the following image:



	1	2
1	100	
2	50	113.3
3	51	108.9
4	52	109.1
5	53	106.5

Save Cal Factors Click to save the cal factor table to a *.csv file.

Use this sensor only Check this box to use this sensor over the entire frequency span of the measurement, even if two sensors are connected to power meter.

Clear this box to allow entry of minimum and maximum frequencies for the sensor. Only ONE of the two sensors can have this box checked. You will be prompted to connect the appropriate sensor during the power calibration.

Minimum Frequency Specifies the minimum frequency range for the sensor when using dual sensors.

Maximum Frequency Specifies the maximum frequency range for the sensor when using dual sensors.

Zero and Calibrate the Power Sensor

For highest accuracy, Zero AND Calibrate the power sensor before measuring data. Follow prompts that may appear.

Zero - If the following settings are 'greyed', Internal or External zeroing is selected automatically based on the power meter/sensor model. Otherwise, select the appropriate type of zeroing to perform, then press **Zero**.

- **Internal Zero** - A switch inside the power sensor removes the sensor from the incident power.
- **External Zero** - Requires that you physically remove the sensor from incident power.

Note: For the U2000 Series USB power sensors

Calibration is NOT available. Select External Zero ONLY when the power to be measured is **below** the specified level. Otherwise, the U2000 series performs internal zeroing automatically when needed. See your power sensor documentation for more details.

- U200xA - below -30 dBm
- U200xH - below -20 dBm
- U200xB - below 0 dBm

If your U2000 power sensor 'hangs' when external zeroing, upgrade the power sensor firmware to Rev. A.01.02.00 or higher to fix this problem.

Note: For the U2020 X-Series USB power sensors

The U2020 X-Series support only internal zeroing. But like the U2000 series, they default to performing zeroing automatically when needed.

Calibrate - Available when the selected sensor has calibration capability. Calibration involves measuring an internal 1 mW source.

- Agilent P-Series sensors and U2020 X-Series USB sensors have an internal reference so you can calibrate them without connecting to a meter's reference port.
- Agilent U2000 USB power sensors do not require calibrating.
- For other sensors, refer to the documentation to determine if it has calibration capability.

Press **Calibrate**, then follow the prompts.

Copy a Source Power Calibration to other Channels

A macro application is now available that copies a Source Power Calibration to other channels. Once downloaded and installed on a PNA, the [macro](#) is automatically configured up. To learn more, click **Help** on the application main dialog. Get the application from <http://na.tm.agilent.com/pna/apps/applications.htm>.

Saving a Source Power Calibration

Because Source Power Cal calibrates source hardware, the calibration data is saved as part of the **Instrument State**, in either a .sta file or a .cst file. This correction is applied to all measurements on the channel that uses the calibrated source. See [Save Instrument State](#).

Reducing Time to Complete a Source Power Calibration

The time required to perform a Source Power Calibration depends on source power, number of points, and number of readings taken. You can reduce this measurement time with the following methods:

- **Reduce number of points before calibration.** You can reduce the number of points before the measurement, then return the number of points to its original value after calibration is complete and correction is ON. The analyzer will perform a linear interpolation, although with some loss in accuracy.
- **Use an Agilent E-Series sensor.** You can obtain 40+ readings per second over GPIB with this type of sensor on the PNA.
- **Increase power to the sensor.** Lower power may have longer settling time with some sensors.
- Check [Use Reference Receiver for Iteration](#).

Receiver Power Calibration

Note: Beginning with PNA firmware rev A.09.30, a Guided Power Cal can be performed during an S-parameter Guided Calibration. [Learn more](#).

Receiver power calibration mathematically removes frequency response errors in the specified PNA receiver, and adjusts readings to the same, or a value offset from, the source power calibration level. It is the same as doing a **Response Cal** or **Data / Memory, (Normalization)** but with the data shifted to the [Cal Power](#) value.

Use Receiver Power Calibration to make very accurate absolute power (amplitude) measurements.

Receiver Power Calibration:

- Is ONLY allowed when making absolute power ([unratioed](#)) measurements.
- Is most accurate when a source power calibration was performed first.
- Applies to all unratioed measurements in the active channel using that receiver.
- Can be saved in a Cal Set and later reapplied to a like measurement.

Interpolation

Like other calibration types, if the original stimulus settings are narrowed, interpolation is applied and **C* Rcvr Pwr** is displayed in the status bar. If the original stimulus settings are made wider, the PNA will turn Receiver Power Correction **OFF**.

If the original settings are restored, then receiver power calibration returns to full correction.

How to perform a Receiver Power Calibration

1. Perform a [Source Power Calibration](#).
2. Set the active measurement to unratioed. [Learn How](#).
3. Connect a THRU line from the source port to the receiver port.
 - When performing a receiver power cal on a reference receiver (source 1 and receiver R1), no connection is necessary as the receiver is internally connected to the source.
 - When the source port and receiver port are the same (receiver A, source port 1), then connect an open or short to get maximum power to the receiver. This practice is not recommended. It is best to use different ports for the source and receiver.
4. Ensure correction for Source Power Calibration is ON as indicated by **Src Pwr Cal** or **Src Pwr Cal*** in the status bar.
5. Start the [Calibration Wizard](#)

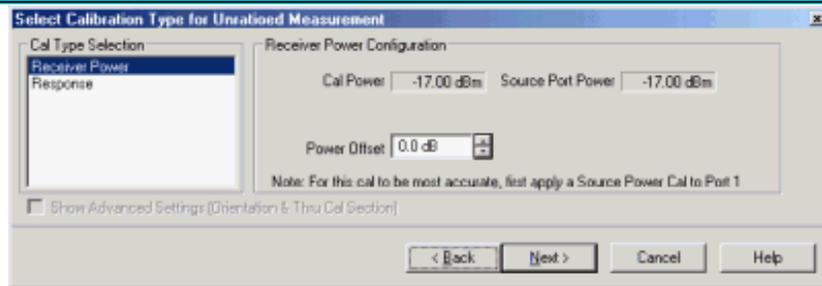
Using front-panel HARDKEY [softkey] buttons

1. Press **CAL**
2. then **[Power Cal]**
3. then **[Receiver Cal]**

Using a mouse with PNA Menus

1. Click **Response**
2. then **Cal**
3. then **Power Cal**
4. then **Receiver Cal**

Select Calibration Type for Unratioed Measurement dialog box help



Cal Type Selection Select **Receiver Power**

Receiver Power Configuration

Cal Power Specifies the power level to be displayed on the measurement when complete. (Source Port Power + Power Offset).

Source Port Power Test port Power set for the measurement. [Learn how to change Test Port Power](#)

Power Offset Allows you to specify a gain or loss (in dB) to account for components you connect between the source and the reference plane of your measurement **AFTER** a source power cal has been performed. Following the calibration, the PNA power readouts are adjusted to the Cal Power value.

Next Click to continue the Calibration Wizard.

Notes:

- When Receiver Power Cal is finished, **'Response'** is displayed in the status bar and correction data is applied to subsequent sweeps. This is done because Receiver Power Cals are essentially Response Cals once they are stored and applied. See Saving a Receiver Power Cal below.
- To turn correction **OFF**, click **Calibration**, point to **Power Calibration**, then set **Receiver Power Correction** to **OFF**.

[Learn more about Receiver Power Cal \(scroll up\).](#)

Saving a Receiver Power Calibration

Beginning with PNA Revision 5.0, Receiver Power Cal is saved to a [Cal Register](#) and optionally to a [User Cal Set](#). It can be applied to measurements in the same way as other Cal Types. Previously, Receiver Power Cal data was saved as part of an Instrument State and was only applied to the

measurement on which it was performed.

[Learn more about Saving PNA files types.](#)

Last modified:

13-Feb-2014	Added U8488 and other series
15-Jan-2014	Added explan. for Max and Min power levels Fixed typos in SPC Options diag Added CF Load and Save Receiver Power Cal annotation changed
23-Oct-2012	Added U2020 series sensors
19-Jul-2012	Support for R&S sensors
12-Jul-2012	Separate zero and cal buttons. Increase Max iterations
14-May-2012	Increased limit for Loss table
1-May-2012	Added power meter msg
23-Apr-2012	Added two notes to PM settings dialog
5-Apr-2012	Added emulation note
28-Jun-2011	Modified First time note
10-Jun-2011	Removed LAN restrictive note
7-Mar-2011	Minor support edits
22-Oct-2010	Add N914x meters
21-Oct-2010	Add Calibrate at multiple power levels
30-Jan-2009	Added USB sensor note
11-Dec-2008	Added tolerance increment.
8-Dec-2008	Added persistence of power loss table Link to characterize Adaptor Macro A.08.33
14-Aug-2008	Clarified a few concepts from BH.
24-Jun-2008	Updated diag image
2-Jun-2008	Clarified Power Loss compensation.
21-Feb-2008	Added 848x note
4-Jan-2008	Added Cal note for USB sensors
30-Oct-2007	Added link to supported Power meters/ sensors
20-Jul-2007	Added USB / LAN support and Apply macro
21-Jan-2007	MX Added UI
14 Sept-2006	MQ Added Receiver-only SPC.

Fixture Simulator

The following features allow you to mathematically add (embed) or remove (de-embed) circuits to, or from, your PNA measurements. The mathematical models are applied to specific ports for all measurements on the channel.

Notes

- Beginning with PNA Rev. A.09.20, the following features are available in [GCA](#), [GCX](#), [Swept IMD](#), [Swept IMDX](#), [Noise Figure](#), and [NFX](#) Apps:
 - [Port Extensions](#) (Not available in Swept IMD or IMDX)
 - [2 Port De-embedding](#)
 - [Port Matching](#)
 - [Port Z Conversion](#)
- All other Fixturing features are available ONLY in a [standard channel](#).
- When fixturing is enabled, all of the enabled fixturing features are applied when [snp files are saved](#).

See Also

- **Procedures:** [To Embed or De-embed?](#)
- [Characterize Adaptor Macro](#) can be used to create S2P files from Cal Sets.
- ["De-embedding and Embedding S-Parameter Networks Using a Vector Network Analyzer" App note](#), for more conceptual information on Fixture Simulation.
- [See an example](#) of how these functions can be used to de-embed unwanted effects of a test fixture, and then mathematically embed the DUT in the circuit in which it is used.

Order of Fixture Operations

First, the following **Single-ended** measurement functions are processed in this order:

1. [Port Extensions](#)
2. [2-Port De-embedding](#)
3. [Port Matching Circuit Embedding](#)

4. [Port Z \(Impedance\) Conversion](#)
5. [4-Port Network \(single-ended\) Embed/De-embed](#)

Then, **Balanced** measurement functions are processed in this order:

6. [Balanced Conversion](#)
7. [Differential](#) / [Common Mode](#) Port Z Conversion
8. [Differential Matching Circuit Embedding](#)

- [Source power compensation](#) is then optionally applied to compensate for the aggregate loss through all enabled fixturing operations.

Notes

- The fixturing operations are applied to the measurement results.
- The order of operations **1 through 4** can be changed using the SCPI command: [CALC:FSIM:SEND:OORD](#). Learn how to send this command from the [GPIB Command Processor Console](#).
- The order of the operations **5 through 8** can NOT be changed.
- In the [PNA data processing chain](#), the Fixture Simulator functions occur at the same time as the **Apply Error Terms** block.

How to select Fixturing Simulator

About Fixturing ON/off

BOTH of the following must occur to turn a fixturing selection **ON**.

EITHER ONE will turn a fixturing selection **OFF**.

1. Check **Fixturing ON/off**
Port Extensions is NOT affected by Fixturing ON/off.
2. Check **Enable** on the individual fixturing selection dialog box.

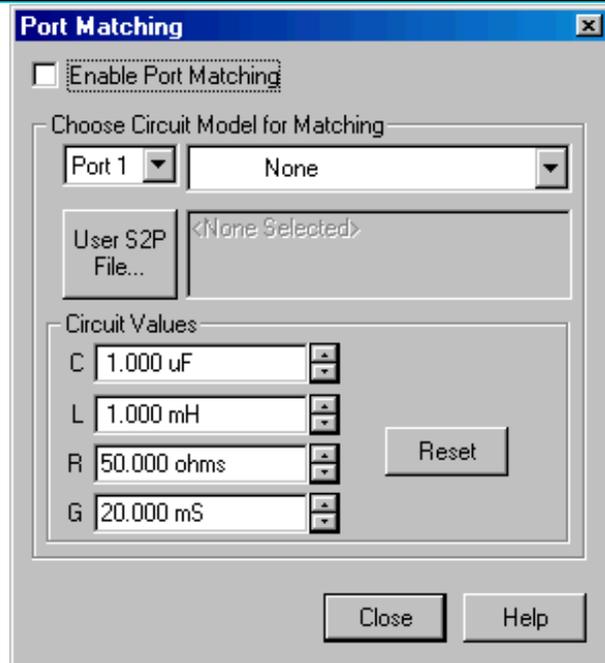
Using front-panel HARDKEY [softkey] buttons

1. Press **CAL**
2. then **[More]**
3. then **[Fixtures]**

Using a mouse with PNA Menus

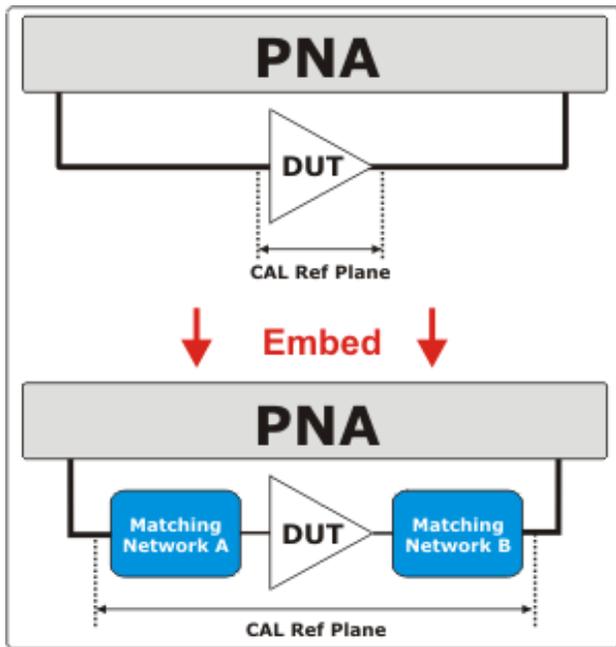
1. Click **Cal**
2. then **More**
3. then **Fixtures**

Port Matching dialog box help



Note: Beginning with PNA Rev. A.09.20, this feature is available to the following [measurement classes](#): [GCA](#), [Swept IMD](#), [Swept IMDx](#), [Noise Figure](#), [NFX](#), and standard (S-Parameter) channels.

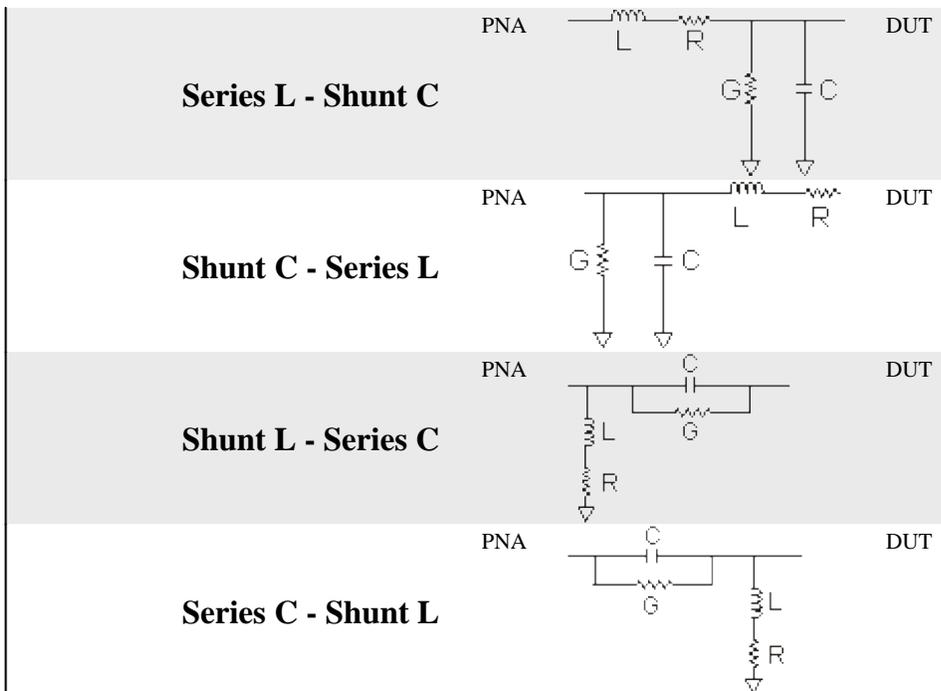
This function specifies a circuit to embed (add) to the measurement results. [See Order of Fixture Operations](#).



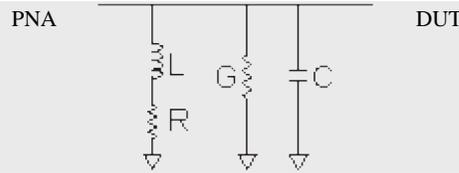
Enable Port Matching Check to apply the settings to the measurement results. Must also enable [Fixturing ON/off](#).

Port - Select Port in which to apply simulation.

Circuit Model for Matching - Choose one of the following that best emulates your fixture at the selected PNA port:



Shunt L - Shunt C



User Defined (S2P File) Load a file that is specified with **User S2P File** button.

None Use no circuit model.

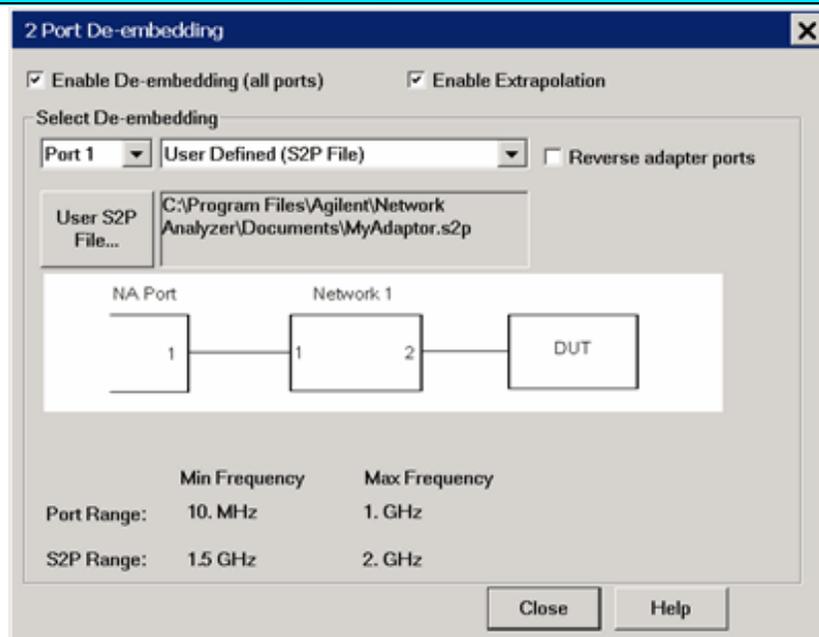
User S2P File Click to specify an S2P file of the circuit model to embed at the selected port. If the normalized impedance value in a recalled User .S2P file is different from the port reference impedance setting of the PNA, the PNA setting is used. [Characterize Adaptor Macro](#) can be used to create S2P files from Cal Sets.

Circuit Values

Capacitance (C), Inductance(L), Resistance(R), Conductance(G) Values for the specific components of the circuit type that models your fixture.

Reset Restores the default values.

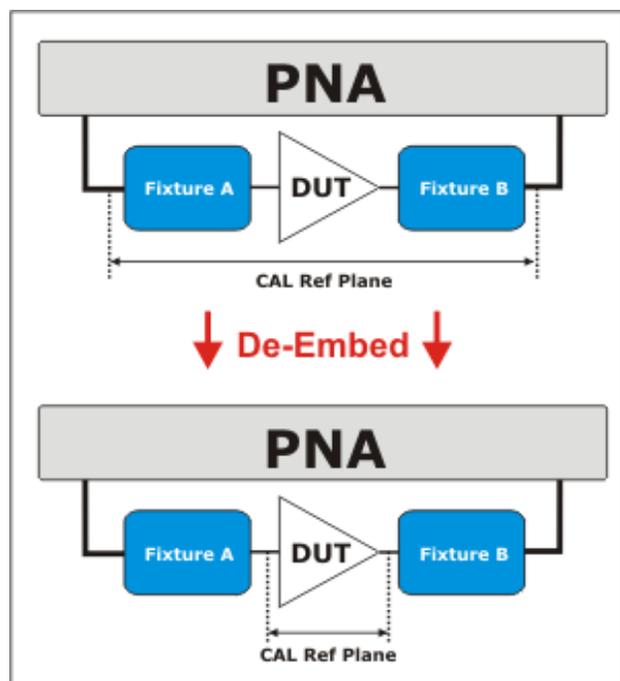
2 Port De-embedding dialog box help



Note: Beginning with PNA Rev. A.09.20, this feature is available to the following [measurement classes](#): [GCA](#), [Swept IMD](#), [Swept IMDx](#), [Noise Figure](#), [NFX](#), and standard (S-Parameter) channels.

De-Embed when you have performed a calibration and then added a fixture (an adapter, an attenuator, a longer cable, etc.) that connects between the Cal reference plane and your DUT. This function **removes** the effects of a component or test fixture from the measurement results.

Note: De-embedding a component with more than 20 dB of loss becomes impractical because of an inability to accurately measure the match of the DUT through such a device.



The de-embedding operation recalls an .s2p file (Touchstone format) which includes the electrical characteristics of a 2-port fixture or device. The file can be in any standard format (real-imaginary, magnitude-angle, dB-angle).

Enable De-embedding Check to apply the settings to the measurement results. Must also enable [Fixturing ON/off](#).

Enable Extrapolation Check to apply a simple extrapolation when the S2P file has a narrower frequency range than the channel. The values for the first and last data points are extended in either direction to cover the frequency range of the measurement. The frequency ranges of both the channel and the S2P file are displayed at the bottom of the dialog.

When extrapolation is necessary and enabled, a message is displayed showing the frequency range to be extrapolated. When extrapolation is necessary and disabled, a message is displayed offering to enable extrapolation.

This setting also causes [4-port Extrapolation](#) to be enabled and disabled.

Port The PNA port to which the recalled de-embedding file is applied.

From the drop-down menu, select **User Defined (S2P File)**.

Reverse Adaptor Ports Check to cause the Fixture/Adapter to be configured with Port 2 connected to the PNA and Port 1 to be connected to the DUT. The image in the dialog reflects that change.

User S2P File Click to specify an existing .S2P file. If the normalized impedance value in a recalled User .S2P file is different from the port reference impedance setting of the PNA, the PNA setting is used. [Characterize Adaptor Macro](#) can be used to create S2P files from Cal Sets.

Port Z (Impedance) Conversion dialog box help



Note: Beginning with PNA Rev. A.09.20, this feature is available to the following [measurement classes](#): [GCA](#), [Swept IMD](#), [Swept IMDx](#), [Noise Figure](#), [NFX](#), and standard (S-Parameter) channels.

This function corrects the measurement and displays the results as if the measurement had been made into the specified impedance value. However, the physical port termination is still approximately 50 ohms.

The specified impedance value is applied to all of the measurements on **ONLY** the active channel.

[See Order of Fixture Operations.](#)

Enable Port Z Conversion Check to apply the settings to the measurement results. Must also enable [Fixturing ON/off](#).

R Real part of the impedance value.

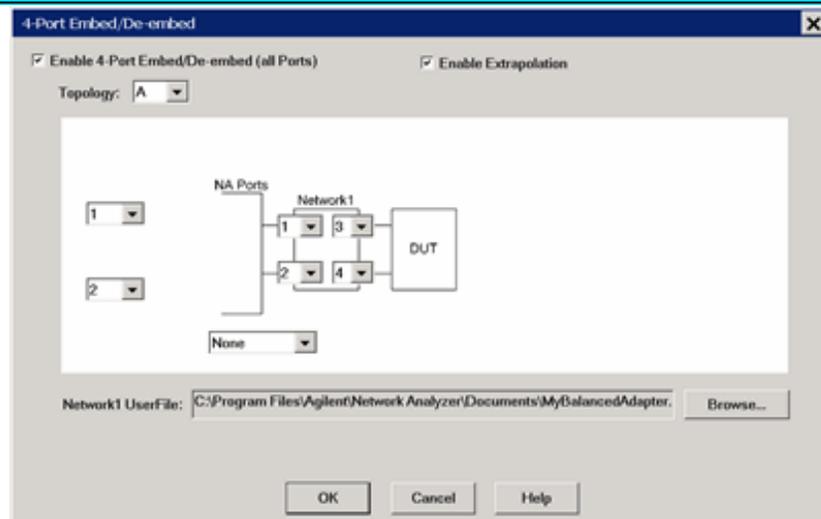
jX Imaginary part of the impedance value.

Close Applies the entries and closes the dialog box

Note: Port Z (Impedance) conversion uses values in the following prioritized order:

1. [Balanced \(Differential or Common Mode\)](#) - if enabled, these values are always used.
2. [Single Port Impedance](#) - if enabled, this value is used if Balanced is not enabled.
3. [System Impedance](#) - if neither balanced or single port is enabled, this value is used.

4-Port Embed/De-embed dialog box help



This function specifies a single-ended 4-port circuit (*.S4P file) to embed (add) or de-embed (remove) from the measurement results. Computation takes place BEFORE Balanced conversion. [See Order of Fixture Operations.](#)

There is a single normalized impedance value for each port in the *.S4P file. This impedance value must match the impedance of the previous Port Z setting, or the PNA port impedance.

The PNA will interpolate if the number of data points that are read is different from the current PNA setting.

Enable 4-Port Embed/De-embed Check to apply the settings to the measurement results. Must also enable [Fixturing ON/off](#).

Enable Extrapolation Check to apply a simple extrapolation when the S4P file has a narrower frequency range than the channel. The values for the first and last data points are extended in either direction to cover the frequency range of the measurement. The frequency ranges of both the channel and the S4P file are displayed at the bottom of the dialog.

When extrapolation is necessary and enabled, a message is displayed showing the frequency range to be extrapolated. When extrapolation is necessary and disabled, a message is displayed offering to enable extrapolation.

This setting also causes [2-port Extrapolation](#) to be enabled and disabled.

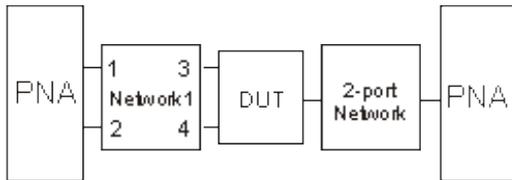
Topology

Select a DUT topology. Refer to the images on the 4-port embed/De-embed dialog box.

- **A** - 2 PNA/DUT Ports
- **B** - 3 PNA/DUT Ports
- **C** - 4 PNA/DUT Ports

Note: Not all possible DUT topologies are addressed with this dialog box.

To embed or de-embed other topologies, use this dialog in conjunction with the [2-port embed/de-embed dialog](#). For example, if you have a 3-port Bal-SE DUT and have networks to de-embed as shown here:



1. In this dialog specify **Topology B**.
2. De-embed the 4-port Network1 on the Balanced input.
3. Use the [2-port dialog](#) to de-embed the 2-port network on the Single-ended output.

NA Ports - Select the PNA Port that is connected to each circuit port.

Network Ports Select the network ports that represent the configuration of the S4P file. By default, ports 1 and 2 are connected to the PNA and ports 3 and 4 are connected to the DUT.

None, Embed, De-embed For Network1 and Network2, select:

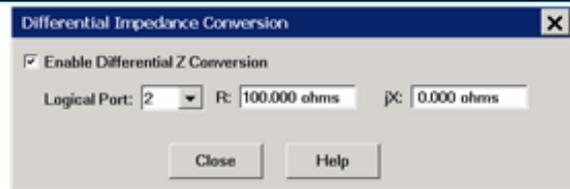
- **None** - The same as disabling.
- **Embed** - Add the specified network circuit to the measurement results. [See 2-port Embed image](#).
- **De-embed** - Remove the specified network circuit from the measurement results. [See 2-port De-embed image](#).

Browse For both Network1 and Network2, navigate to find the .*S4P file to embed or de-embed.

OK Applies the changes and closes the dialog box.

Cancel Does NOT apply the changes and closes the dialog box.

Differential Impedance Conversion dialog box help



This function sets the Differential impedance value for each balanced port.

The default value for **R**: is the SUM of the impedance values for both ports that make the logical port. If [Port Z Conversion](#) is not enabled, then [System Z0](#) values for both ports are summed.

[See Order of Fixture Operations.](#)

Enable Differential Z Conversion Check to apply the settings to the measurement results. Must also enable [Fixturing ON/off](#).

Logical Port Select the logical (balanced) port to receive impedance value. To see logical port numbers, see the [measurement topology](#).

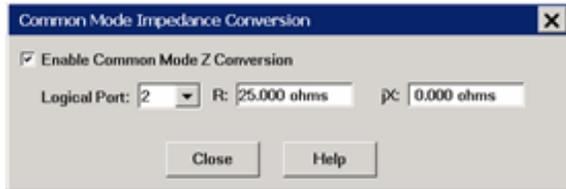
R Real part of the impedance value.

jX Imaginary part of the impedance value.

Close Closes the dialog box.

[See note about Port Impedance priority.](#)

Common Mode Impedance Conversion dialog box help



This function sets Common Mode Impedance value for each balanced port.

The default value for **R**: is calculated as follows.

$$(Z1 * Z2) / (Z1 + Z2)$$

Where ports 1 and 2 comprise the logical port:

Z1 = the Port Impedance values for port 1

Z2 = the Port Impedance values for port 2

If [Port Z Conversion](#) is not enabled, then [System Z0](#) values for port 1 and 2 are used in the calculation.

[See Order of Fixture Operations.](#)

Enable Common Mode Z Conversion Check to apply the settings to the measurement results. Must also enable [Fixturing ON/off](#).

Logical Port Select the logical (balanced) port to receive impedance value. To see logical port numbers, see the [measurement topology](#).

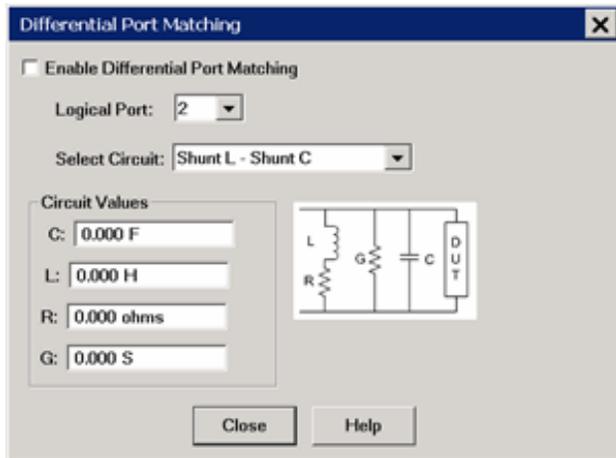
R Real part of the impedance value.

jX Imaginary part of the impedance value.

Close Closes the dialog box.

[See note about Port Impedance priority.](#)

Differential Port Matching dialog box help



This function allows the embedding of a differential matching circuit at a balanced port.

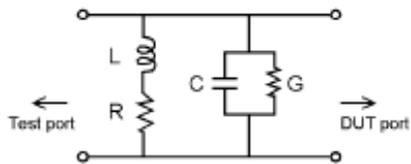
[See Order of Fixture Operations.](#)

Enable Differential Port Matching Check to embed the selected matching circuit to the measurement results. Must also enable [Fixturing ON/off](#).

Logical Port Choose [Logical DUT port](#) to receive the selected matching circuit. To see logical port numbers, see the [measurement topology](#).

Select Circuit Select a matching circuit. Choose from:

- **Shunt L - Shunt C** Predefined circuit.



Circuit Values Choose from:

- **C** Capacitance value
- **G** Conductance value
- **L** Inductance value
- **R** Resistance value
- **User defined** Select an *.S2P file that represents the matching circuit. Then click **Browse** to navigate to the *.S2P file.

Note: For the *.S2P file:

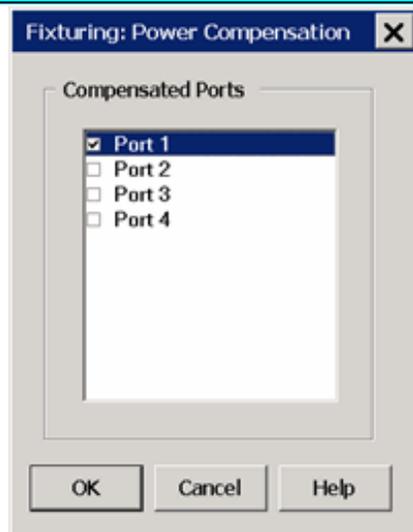
Port 1 of the circuit is assumed to be connected to the PNA

Port 2 of the circuit is assumed to be connected to the DUT.

- **None** No embedded circuit on selected port.

Close Closes the dialog box.

Power Compensation dialog help



This function adjusts the source power at the specified port to compensate for the combined amount of gain or loss through ALL enabled fixturing operations. Use this function to set the power level at the DUT input.

For example:

- Your DUT requires a fixture on the input port which is connected to PNA port 1.
- The fixture description (such as an S2P file at the [2-port De-embed function](#)) indicates the fixture has approximately 2 dB of loss across the frequency span.
- You set source power to 0 dBm. But you want 0 dBm at the DUT input (the fixture output).
- Check Power Compensation on Port 1 and enable [Fixturing](#).
- Power Compensation causes the source power to be increased by approximately 2 dB so that the power at the fixture output plane will remain at 0 dBm.

Power Compensation affects all measurements in the channel.

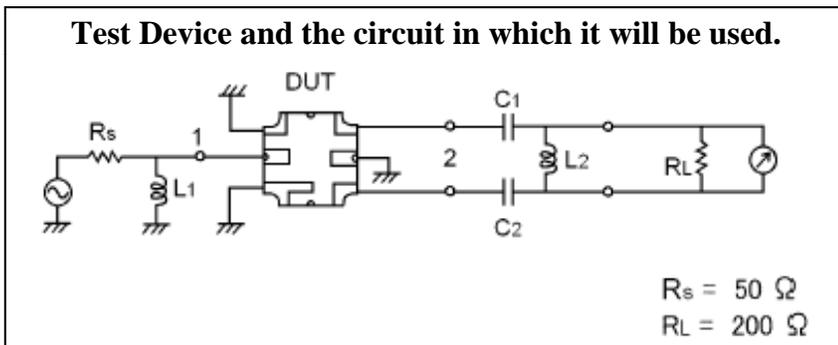
Enable [Fixturing](#) to use Power Compensation.

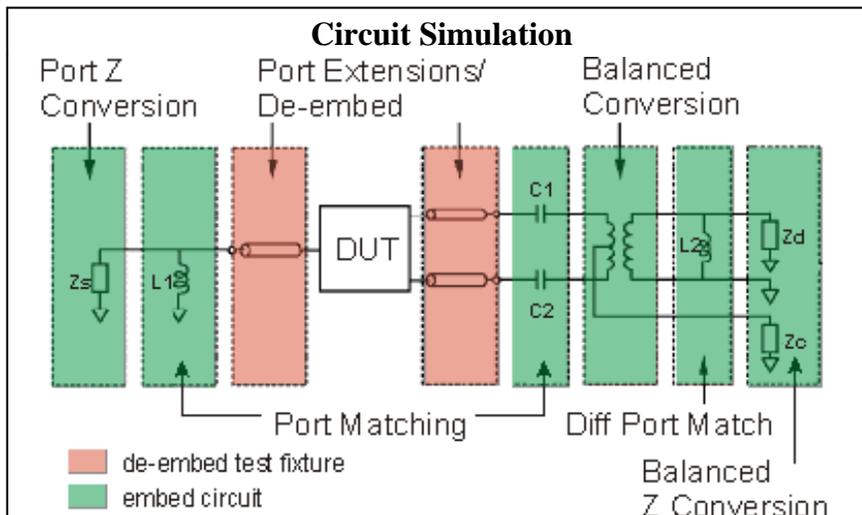
Note: Use caution when applying power compensation. Always test your setup without a DUT in a place. If you are using S2P files, [Recall](#) your S2P file into the PNA so you can verify that the device your S2P file describes is what you intended it to be. It is too easy to misalign data in S2P files if they are constructed by hand.

Fixture Simulator Example

The following example shows a DUT and the matching circuit with which the DUT will be used in its intended application. When the DUT is tested in a high-volume manufacturing environment, multiple test fixtures are often required. The most accurate way to test the DUT and ensure measurement consistency between the different test fixtures is to use a simple, repeatable, test fixture without the actual matching elements.

To get the desired performance data, the parasitic effects of the fixture must first be removed (de-embedded) from the measured data. Then a perfect "virtual" matching circuit must be simulated and added mathematically (embedded) to the corrected, measured data. The result is an accurate display of the DUT as though it was actually tested with a physical matching circuit, but without the uncertainties of using real components.





This diagram does NOT refer to the order in which operations are performed.

[See Order of Fixture Operations.](#)

1. [Create a balanced measurement](#) using single-ended to balanced (SE-Bal) [topology](#). Include all relevant measurement settings (IFBW, number of points, and so forth). Once the measurement is created and calibrated, the measurement parameter can be easily changed. For example, Sdd22 to Sds21.
2. Calibrate the measurement at the point where the simple test fixture is connected to the PNA. Use accurate calibration standards and definitions.
3. Remove the effects of the three uncalibrated transmission lines of the simple test fixture. This can be done in several different methods. The easiest is to use manual or automatic [Port Extensions](#) to move the calibration reference plane to the DUT. This removes the electrical length and loss of the fixture's transmission lines, but does not account for fixture mismatch. Another method is to de-embed previously-created *.S2p files of the 3 transmission lines. The files can be created using external ADS modeling software. Another alternative is to create the *.S2P files by independently measuring all 3 ports of the test fixture and [saving the results of each to an S2P file](#).
4. With the test fixture connected to the PNA and a DUT inserted, the measurement results now appear as though calibration was performed at the connections to the DUT, and the device was measured in a 50-ohm single-ended test environment. The following steps will cause the results to reflect the performance of the device as though the device is embedded in the circuit in which it will be used.
5. Port 1 of the device is a single-ended port and sees a source impedance the same as the PNA system impedance, so no change is required. However, if R_s were a value other than 50 ohms, [Port 1 Impedance Conversion](#) would be used to simulate the different impedance.
6. [Port Matching](#) is used to simulate L1 inductance. Select any of the Shunt L circuits to embed (add) to the measurement results. Enter the value of L and R. The C and G values can be entered as 0 (zero).
7. [Port Matching](#) is used to simulate C1 and C2 capacitance. For both port 2 and port 3, select any of the **Series C** circuits to embed (add) to the measurement results. Enter the value of C and G. The L and R values can be entered as 0 (zero).

8. [Balanced Conversion](#) mathematically simulates the measurement in balanced mode.
 9. [Differential Port Matching](#) is used to simulate L2 inductance. Select Shunt L- Shunt C and enter the inductance / resistance value. The C and G values can be entered as 0 (zero).
 10. Finally, [Differential Z Conversion](#) is used to simulate a circuit termination of 200 ohms. If you are making Common Mode measurements, specify [Common Mode Z Conversion](#).
-

Last modified:

30-Jan-2013	Added de-embed 'impractical' note
29-Oct-2012	Minor edit to ordering
2-May-2012	Edited 4-port embed notes
12-Nov-2010	Added extrapolation and reverse (A.09.33)
12-Apr-2010	Updated for A.09.20
11-Nov-2009	Add SNP note
2-Feb-2009	Reorder and add CAN change first four
29-Sep-2008	Added note about standard channel
3-Sep-2008	Removed legacy content
Sept 12, 2006	Added link to programming commands

Port Extensions

Port extensions allow you to electrically move the measurement reference plane after you have performed a calibration.

Note: Beginning with PNA Rev. A.09.20, this feature is available to [GCA](#), and standard (S-Parameter) channels.

- [Why and How to use Port Extensions](#)
- [Manual Port Extensions Procedure](#)
- [Port Extensions dialog and Toolbar](#)
- [Step Size dialog](#)
- [Automatic Port Extension dialog](#)

See Also

[PNA Data Flow Map](#)

[Fixture Compensation features](#)

[Phase Accuracy](#)

[Comparing the PNA Delay Functions](#)

Other Calibration Topics

Why use Port Extensions

1. You are unable to perform a calibration directly at your device because it is in a test fixture. Perform a calibration at a convenient place, then use port extensions to compensate for the time delay (phase shift), and optionally the loss, caused by the fixture.
2. You have already performed a calibration, and then decide that you need to add a length of transmission line in the measurement configuration. Use port extensions to "tell" the analyzer you have added the length to a specific port.

Important Note: Port Extensions and PNA Data Flow

See [PNA Data Flow diagram](#)

Normally, Port Extensions are applied to individual S-parameters in the **Phase Correction** process and only applies to displayed S-parameters.

However, when [Fixturing](#) is ON or when making a [Balanced Measurement](#), Port Extension compensation is applied in the **Apply Error Terms** process which affects ALL S-parameters, whether displayed or not. This allows all underlying S-parameters to have proper extensions applied.

Therefore, when using Port Extensions with features that require more than a single S-parameter (such as k-factor in equation editor), do one of the following:

- Enable [Fixturing](#) - Individual Fixturing features are NOT required to be enabled.
- Use [8510 Mode Data Processing](#).

When Port Extension compensation is applied in the **Apply Error Terms** process, after a [Data-to-Memory](#) operation has been performed, further changes to Port Extensions settings will NOT be applied to the Memory trace.

How to use Port Extensions

- If you know the **electrical length** of the fixture or additional transmission line, enter the value directly to the **Time** setting.
- If you know the **physical length** of the fixture or additional transmission line, enter the value directly to the **Distance** setting.
- If you do **NOT** know either the electrical or physical length of the fixture or additional transmission line, you must be able to connect an OPEN or SHORT to the new reference plane - in place of the DUT. In most cases, removing the DUT will leave a suitable OPEN at the new reference plane.
- Port Extensions can then be added manually (as follows), or by using [Automatic Port Extensions](#).

Manual Port Extensions Procedure

1. Select a calibrated S11 measurement.
2. Select Phase format.
3. With an OPEN or SHORT at the calibration reference plane, verify that the phase across the frequency span

is at or near zero.

4. Connect the fixture or added transmission line and attach an OPEN or SHORT in place of the DUT. In most cases, removing the DUT will leave a suitable OPEN at the new reference plane. On the Port Extension toolbar or dialog, increase either **Time** or **Distance** until the phase response is flat across the frequency span of interest.
5. If you know the loss of the additional transmission line, enter the [Loss Compensation](#) values using either one or two data points.

Note: Most OPEN and SHORT standards have delay. Therefore, adjusting delay with this method results in a delay equal to two times the delay of the OPEN or SHORT.

How to access Port Extensions settings

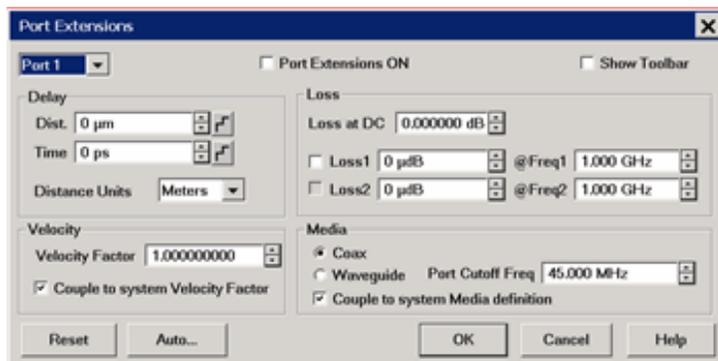
Using front-panel HARDKEY [softkey] buttons

1. Press CAL
2. then [Port Extensions]

Using a mouse with PNA Menus

1. Click **Response**
2. then **Cal**
3. then **Port Extension**

Programming Commands



Port Extensions dialog and Toolbar help

Note: Beginning with PNA Rev. A.09.20, this feature is available to the following [measurement classes](#): [GCA](#), [Noise Fixture](#), [NFX](#), and standard (S-Parameter) channels.

Port extensions settings affect all measurements on the active channel that are associated with a particular port.

[Learn Why and How to use Port Extensions](#) (scroll up).

Port Extension Turns ON and OFF port extensions on all ports.

Port Select a PNA port for delay and loss values. Port Extensions settings affect ALL measurements on the active channel that are associated with a particular port.

Show Toolbar Check to show the Port Extensions toolbar. The toolbar allows you to make adjustments to the port extensions while showing more of the PNA screen. Beginning with PNA rev A.08.50, this is the only way to show or hide the toolbar.

Delay settings

Enter delay in either Distance or Time by entering a value or clicking the up/down arrows. Click  to start the [Step Size](#) dialog.

Time The amount of port extension delay in time. Enter a positive value.

Distance The amount of port extension delay in physical length. Enter a positive value.

Distance Units (Dialog ONLY) Select from Meters, Inches, or Feet. The Step Size setting will not change automatically. [Learn more.](#)

Loss Compensation

The following settings allow the entire frequency span to be corrected for loss.

Loss at DC Offsets the entire frequency span by this value. Loss1 or Use1 must also be checked. To compensate for loss at DC, enter a positive value which causes the trace to shift in the positive (up) direction.

Loss @Frequency Check the box, and enter values for Loss and Frequency

When **Loss1** or **Loss1/Loss2** are used, a curved-fit algorithm is used as follows:

Loss1 ONLY:

$$\text{Loss}(f) = \text{Loss1} * (f/\text{Freq1}) ^ 0.5$$

Loss1 and Loss2:

Set the lower frequency to Loss1, and the higher frequency to Loss2.

$$\text{Loss}(f) = \text{Loss1} * (f/\text{Freq1}) ^ n$$

Where:

$$n = \log_{10} [\text{abs}(\text{Loss1}/\text{Loss2})] / \log_{10} (\text{Freq1}/\text{Freq2})$$

Note: abs = absolute value

Velocity

Velocity Factor For each port, sets the velocity factor that applies to the medium of the device that was inserted after the measurement calibration. The value for a polyethylene dielectric cable is 0.66 and 0.7 for PTFE dielectric. 1.0 corresponds to the speed of light in a vacuum.

Couple to system Velocity Factor When unchecked, the Velocity Factor is set for only the specified port and only for Port Extensions. When checked, sets the Velocity Factor for all ports. In addition, changing this value also changes this setting for the [Electrical Delay](#) and [Time Domain Distance Marker](#) features.

Media

For each port, select the media of the added transmission line or fixturing.

Coax Select when the fixture or added transmission line is coax. Also specify the velocity factor of the coax.

Waveguide / Cutoff Frequency Select when the fixture or added transmission line is waveguide. Also enter cutoff (minimum) frequency of the waveguide.

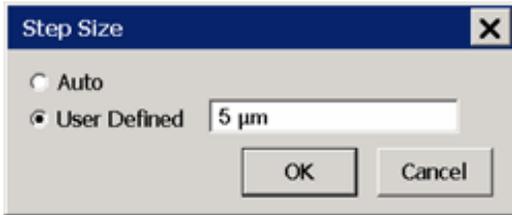
Note: when using a Waveguide cal Kit, set [System Z0](#) to 1 ohm before calibrating.

Couple to system Media Definition. When unchecked, the Waveguide Cutoff Frequency is set for only the specified port and only for Port Extensions. When checked, sets the Waveguide Cutoff Frequency for all ports. In addition, changing this value also changes this setting for the [Electrical Delay](#) feature.

Reset All port extensions settings are changed to preset values. The Port Extension ON / OFF state is NOT affected.

Auto Ext. Starts the [Automatic Port Extensions](#) dialog box

Note: Individual receiver port extensions (A,B, and so forth) can no longer be set. (Sept. 2004)



Step Size dialog box help

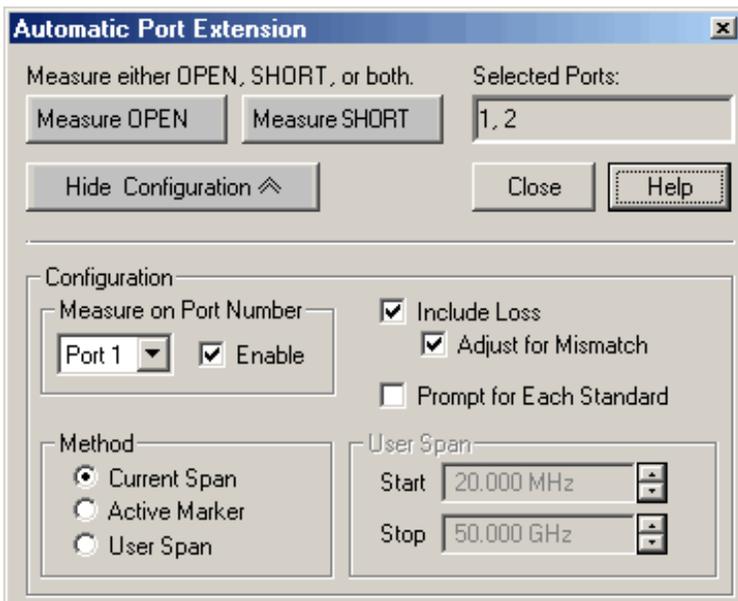
Changes the step size that occurs when the Time or Dist up/down arrows are pressed on the Port Extension toolbar. The Units for step size are changed on the Port Extension dialog.

Auto Step Size is set to the default value.

User Defined Enter a step size value, then click OK.

This value remains the same when the units are changed. For example if a step size of 12 is entered on this dialog, then you change the units from Inches to Feet, the step size of 12 inches becomes 12 feet, not 1 feet. Therefore, change the units first, then set the step size.

Learn about [Port Extensions](#) (scroll up)



Automatic Port Extension dialog box help

Automatic Port Extension AUTOMATICALLY performs the same operation as [Manual Port Extension](#). By connecting a SHORT or OPEN, the reference plane is automatically moved to the point at which the standard is connected. In addition, Automatic Port Extension will optionally measure and compensate for the loss of the additional transmission line.

Auto Port Extension is NOT available when:

- Sweep type is set to power sweep
- Frequency Offset is ON
- Media is set to Waveguide

Note: Turn OFF [Equations](#) that may exist on the active trace when using Automatic Port Extensions.

Auto Port Extensions Procedure

1. Connect the added transmission line or fixture. Attach an OPEN or SHORT to all affected ports at the new reference plane. In most cases, removing the DUT will leave a suitable OPEN at the new reference plane.
2. On the Port Extension toolbar, click **Auto Port Ext.** Click **Show Configuration** to make additional settings.
3. Click **Measure** to perform the port extension calculations. The resulting delay and loss settings are entered into the port extension toolbar. These settings are saved with Instrument Save or you can manually record the values and enter them again when required.

Settings

Measure either OPEN, SHORT, or both Press a button to make the measurement of the reflection standard.

Measure either OPEN or SHORT depending on which is most convenient. An ideal OPEN and SHORT, with zero loss and delay, is assumed. Therefore, accuracy is most affected by the quality of the standard. In most cases, removing the DUT will leave a suitable OPEN at the new reference plane. When measuring both OPEN and SHORT standards, the average of the two is used and will slightly improve accuracy.

Selected Ports Indicates the ports that currently have automatic port extension enabled. By default, ALL PNA ports are enabled. To disable a port, see **Measure on Port Number** below.

Note: Port Extensions settings affect ALL measurements on the active channel that are associated with a particular port.

Show/Hide Configuration Press to either show or hide the following configuration settings in the dialog box.

Measure on Port Number

Select port number to enable or disable automatic port extension.

Enable Check to enable the specified port. All enabled ports will have their reference plane automatically adjusted after performing Automatic Port Extension.

Include Loss Check to automatically measure the loss in the additional transmission line and apply compensation. To calculate loss compensation, frequencies at 1/4 and 3/4 through the frequency range are usually used as Freq1 and Freq2 values. [Learn more about Loss Compensation.](#)

Adjust for Mismatch Only available when **Include Loss** is checked. During the measurement of the OPEN or SHORT standard, mismatch could cause ripple in the magnitude (loss) response. The [Loss compensation curved-fit algorithm](#) allows half of the ripple to be positive and half negative. When measuring low-loss devices, it is possible that some magnitude responses could become slightly positive, indicating gain rather than loss.

Check - Offsets the trace to cause all of the data points to be at or below zero.

Clear - Most accurate application of the curve-fit calculation, but allows positive responses.

Prompt for Each Standard Check to invoke a prompt when the Measure OPEN or SHORT button is pressed. The prompt will indicate which standard to connect to which port.

Method

Select the span of data points which will be used to determine correction values for phase and loss (optional). If a portion of the current frequency span does not have flat or linear response, you can eliminate this portion from the calculations by using a reduced User Span.

To calculate loss compensation, Current Span and User Span methods usually use frequencies at 1/4 and 3/4 through the frequency range as Freq1 and Freq2 values. See [Loss Compensation](#) to learn more about how loss is calculated.

Current Span Use the entire frequency span to determine phase and loss values.

Active Marker Use only the frequency at the active marker, and one data point higher in frequency, to calculate phase and loss values. If a marker is not present, one will be created in the center of the frequency span.

User Span Use the following User Span settings to determine phase and loss values.

User Span

Start Enter start frequency of the user span.

Stop Enter stop frequency of the user span.

Learn about [Port Extensions](#) (scroll up).

Last modified:

1-Jun-2011	Edited data flow note
12-Apr-2011	Added note about data flow
18-May-2009	Added note about equations
5-Feb-2009	Added many features (A.08.50)
3-Sep-2008	Removed legacy content
9/12/06	Added link to programming commands

Characterize Adaptor Macro

This external [Macro](#) application creates an S2P file that models a device such as an adaptor, the input OR output side of a test fixture, or an on-wafer probe head. This is done by calculating the four S-parameters of the device from two 1-port calibrations. Such S2P files can be used for embedding or de-embedding the device from S-parameter measurements and FCA calibrations.

This application, along with the FCA [Embed/De-embed feature](#), can be especially useful when performing FCA calibrations.

- An SMC calibration requires a [power meter measurement](#) at the port 1 reference plane. This could be very difficult in on-wafer applications where the measurement reference plane is at the tip of a probe. This macro, in conjunction with the FCA Embed/De-embed feature, enables you to model the probe and connect the power sensor at the coax connector where the probe connects.
- Likewise, a VMC calibration requires that a [calibration mixer](#) be used for the Thru standard. Again, this can be very difficult in on-wafer applications where the measurement reference plane is at the tip of a probe. This macro, in conjunction with the FCA Embed/De-embed feature, enables you to model the probe and connect the calibration mixer at the coax connectors where the probe connects.

New: Characterize Adaptor Macro - Version A.02.22

- Saves Forward (measurements) Only
- This macro update requires PNA firmware version A.09.50 or higher

For S-Parameter measurements:

- [To Embed or De-embed](#)
- [Procedures](#)

How to start the Characterize Adaptor Macro

Using front-panel HARDKEY [softkey] buttons

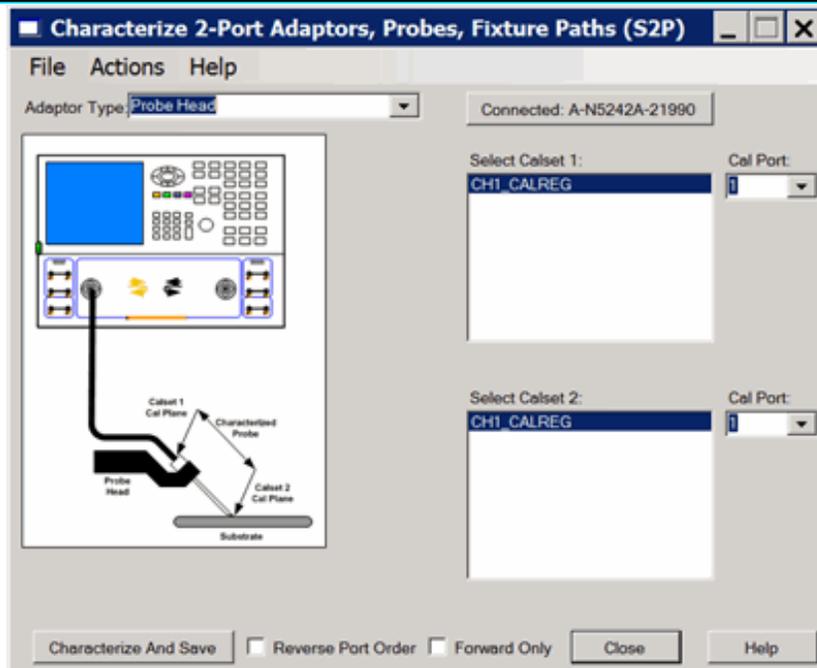
1. Press **MACRO**
2. then [**AdaptorChar**]

Using a mouse with PNA Menus

1. Click **Utility**
2. then **Macro**
3. then **AdaptorChar**

Programming Commands

Characterize 2-Port Adaptors, Probes, Fixture Paths (S2P) dialog box help



Important Notes

- The device to be characterized (probe, adaptor...) MUST be reciprocal ($S_{21} = S_{12}$).
- Two 1-port cals must be performed and saved to Cal Sets BEFORE using the Characterize Adaptor application.
- The frequencies and number of points of the two Cal Sets MUST be identical.
- **CRITICAL:** The calculations that are performed to create the S2P file require that **Calset 1** ALWAYS be from the side closest to the PNA and **Calset 2** ALWAYS be from the other side of the device. If your application that uses the resulting S2P file requires that the ports be reversed, check **Reverse Port Order**. Or learn how to [reverse the port order on an existing S2P file](#).
- The majority of this topic describes the characterization of a 2-port device from two 1-port cal sets. However, you can also use the cal sets from two 2-port calibrations, or one 2-port and two 1-port calibrations. See procedures for both.

[Learn more about the Characterize Adaptor Macro.\(scroll up\)](#)

Connected <PNA host name> The two 1-port Cal Sets can reside in another PNA. Click to connect to another PNA that is DCOM configured. [Learn how to configure DCOM](#).

Adaptor Type Select the type of device to be characterized.

Note: The image that appears in the macro does not influence the calculations. It only appears to help you visualize the measurement reference plane of the Cal Sets.

Select Calset 1 and Calset-2 Select a 1-port Cal Set from each list. Although all Cal sets are listed, only the Cal Sets that have error terms to satisfy a 1-port calibration may be used.

Cal Port Select the port within the selected Cal Set which represents the modeled device. The Cal Ports must be the same for both selected Cal Sets.

Characterize and Save Calculates four S-parameters, then invokes the [Save As dialog](#) with [S2P file type](#). This button is not available until valid Cal Sets and Cal Ports are selected.

Reverse Port Order By default, the calculations that are performed to create the S2P file require that **Calset 1** ALWAYS be from the side closest to the PNA and **Calset 2** ALWAYS be from the other side of the device. Check **Reverse Port Order** to cause the ports to be reversed in the resulting S2P file.

Forward Only The resulting S2P file will contain only S11 and S21 (forward measurements). This is useful for Enhanced Response calibration / de-embedding.

Reverse S2P

Click **Actions**, then **Reverse S2P File**

This action causes ports to be reversed on an existing S2P file. For example:

- The data for S11 becomes the data for S22 and vice versa.
- The data for S21 becomes the data for S12 and vice versa.

The resulting file is written in the standard [PNA S2P file format](#).

1. Navigate to the S2P file to be reversed.
2. Navigate to where the new reversed S2P file will be saved. By default, the file is saved to the same folder as: [old filename]_Reversed.s2p.
3. Click **OK**.

Write to Power Loss Table

Click **Actions**, then **Write to PNA Pwr Loss Table**.



Power Loss Table Setup dialog box help

Loads the S2P Frequency / Loss pairs into the PNA [Power Loss Compensation table](#) to compensate for losses that occur when using the device to connect a power sensor to the measurement port during a Source Power Cal.

Note: PNA firmware revisions prior to A.08.33.07 ONLY support up to 100 segments in the power loss table. Therefore, to save data to the power loss table in one of the earlier firmware revs, then the Cal Sets that created the S2P file must contain 100 data points or less. Revision A.08.33.07 and above allows any number of segments.

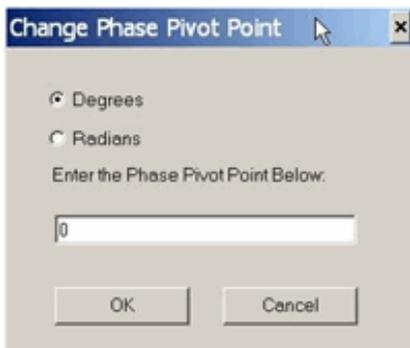
Use Selected Calsets Computes the transmission loss of the fixture based on the selected Cal Sets. This choice is NOT available until two valid Cal Sets are selected.

Use S2P File Uses the S21 data in an existing S2P file to build the PNA's power loss table. Select, then click ... then navigate to the S2P file, then click **OK**.

Note: In the PNA Power Loss Compensation table, loss is expressed as a positive number. The macro assumes that any negative S21 value in the S2P file is a loss and therefore multiplies the S21 values in the file by -1 to express that value as a positive number. This ensures proper handling of the offset during a source power cal.

Change Phase Pivot Point

Click **File**, then **Set Phase Pivot Point**.



Change Phase Pivot Point dialog box help

Before the A.01.50 revision (see Help, About) the Characterize Adaptor macro projected the phase of S21 at DC to cross the X-axis between 0° and -180°, based on the delay of the adapter. For most adapters the value is 0°.

However, when characterizing electrically long cables, cables with significant mismatch, or high noise in the measurements, it is possible that the projection of phase goes above 0°. This results in a 180° phase difference between the results computed by the macro versus the results you might get by measuring the same adapter with a 2-port calibration.

This revision allows you to adjust the center of the 180° phase pivot window . The default value 0° yields a window between +90° to -90° which should be adequate for the majority of adapters.

Enter the value in degrees or radians. This value remains while the application is running.

Restarting the application will result in the Phase Pivot Point to revert back to 0° , the default value.

To Embed or De-embed S-Parameter Measurements

- For FCA measurements, see [FCA Embed or De-embed](#).
- In this section, the term “adapter ” can mean any type of 2-port device and its associated S2P file.

To make an accurate measurement, the setup configuration during the DUT measurement **MUST** exactly match the setup configuration during Calibration. In other words, if you calibrate with an adapter, you must also measure the DUT with the adapter.

However, the PNA provides some flexibility by allowing you to ‘**Virtually**’ add (embed) or remove (de-embed) an adapter from the measurement. Knowing how to do this can be confusing.

- To perform a calibration **WITHOUT** the adapter, but make DUT measurements **WITH** the adapter, then **De-embed** (remove) the adapter from the DUT measurement.
- To perform a calibration **WITH** the adapter, but make DUT measurements **WITHOUT** the adapter then **Embed** (add) the adapter during the DUT measurement.

Procedures

- [Create an S2P file using Characterize Adaptor Macro](#)
- [De-embed the S2P file from DUT measurement](#)
- [Embed the S2P file in DUT measurement](#)

- [De-Embedding a Fixture that has a THRU Standard](#)
- [De-Embedding a Fixture with No THRU Standard](#)

Create an S2P file using the Characterize Adaptor Macro

1. Configure your PNA measurement (frequency span, power level, IF bandwidth, and number of points).
2. Perform a 1-port SmartCal at the reference plane. Save the cal to a User Cal Set using a descriptive name (for example, **Ref Plane**).
3. Connect the adapter to be characterized at the reference plane.
4. Perform another 1-port SmartCal at the end of the adapter. Save it to a User Cal Set using a different descriptive name (for example, **Adapt End**).
5. [Start the Characterize Adaptor Macro](#).
6. In the **Select Calset1** field of the dialog box, select the Cal Set for the reference plane (from step 2 above).
7. In the **Select Calset2** field of the dialog box, select the Cal Set for the end of the adapter (from step 4 above).
8. Click **Characterize and Save**. In the resulting dialog box, enter the .S2P file name and location.
9. Click **Close**.

De-embed the adapter (S2P file) from subsequent S-parameter measurements

See [Fixture De-embedding](#)

Note: Subsequent measurements must have the same or smaller frequency range (within the Start / Stop frequencies) as that of the S2P file.

1. Perform a 2 port SOLT calibration **without** the adapter/fixture.
2. Select 2-port De-embedding: click **Response**, then **Cal**, then **More**, point to **Fixtures**, then click **2 port De-embedding**.
3. Select the Port to add the adapter to, then select **User Defined (S2P file)**.
4. Click **Use S2P file** and select the S2P file created using the Characterize Adaptor macro.
5. Check **Enable De-embedding**, then click **Close**.
6. Enable Fixturing: click **Response**, then **Cal**, then **More**, point to **Fixtures**, then click **Fixturing on/OFF**.
7. **Sim** appears in the Status Bar to indicate that Fixture Simulation is ON.

Embed the adapter (S2P file) into subsequent S-parameter measurements

The adapter (S2P file) can also be a matching network. [Learn more.](#)

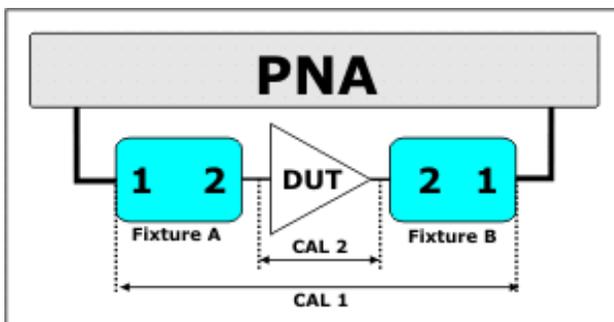
1. Perform a 2 port SOLT calibration **including** the adapter. Note the port number on which the adapter is calibrated.
2. Select Port Matching: click **Response**, then **Cal**, then **More**, point to **Fixtures**, then click **Port Matching**
3. Under Choose Circuit Model for Matching, select the Port that the adapter was on during calibration, then select **User Defined (S2P file)**.
4. Press **Use S2P File** and navigate to the S2P file created using the Characterize Adaptor macro.
5. Check **Enable Port Matching**, then click **Close**.
6. Enable Fixturing: click **Response**, then **Cal**, then **More**, point to **Fixtures**, then **Fixturing on/OFF**.
7. **Sim** appears in the Status Bar to indicate that Fixture Simulation is ON.

De-Embedding a Fixture that has a THRU Standard

A test fixture is generally regarded as a single 'bed' in which a DUT is placed. However, for modeling purposes such as this, it is separated into two circuits: Fixture A on the input of the DUT, and Fixture B on the output.

Use this procedure to perform calibrations **WITHOUT** the test fixture while making measurements **WITH** the test fixture. A calibration is performed once **WITH** the test fixture, and then again as it wears with use and electrical performance changes. The fixture is de-embedded from subsequent measurements to match the regular calibrations that are performed without the fixture.

If you have a THRU standard for your test fixture, you can perform a full 2-port calibration in the fixture, and from that create the required S2P files for de-embedding.



1. Perform a full 2-port **CAL 1** at the connections of the PNA to the fixture as shown above. Save to **MyCalSet1**.
2. Perform a full 2-port **CAL 2** where the DUT is inserted (reference plane). Save to **MyCalSet2**.

Follow the [Create an S2P file](#) procedure, beginning with step 6, using the following selections:

1. Create #1 S2P file:
 1. For CalSet1, choose **MyCalSet1** and select **CalPort=1**
 2. For CalSet2, choose **MyCalSet2** and select **CalPort=1**
 3. Save to **FixtureA.s2p**
2. Create #2 S2P file:
 1. For CalSet1, choose **MyCalSet1** and select **CalPort=2**
 2. For CalSet2, choose **MyCalSet2** and select **CalPort=2**
 3. Save to **FixtureB.s2p**

Follow steps in [To De-embed the adapter...](#)

Perform these steps TWICE; once for each of the following S2P files:

1. For PNA Port 1, select **FixtureA.s2p**
2. For PNA Port 2, select **FixtureB.s2p**

De-Embedding a Fixture with No THRU Standard

This procedure is a slight modification of the above. Cal 2 is performed from two 1-port calcs when a THRU standard for the fixture is not readily available.

1. Perform a full 2-port **CAL 1** at the connections of the PNA to the fixture as shown above. Save to **MyCalSet1**.
2. **CAL 2** is performed using two 1 port calcs
 - **Cal2A** at the Fixture A / DUT plane. Save to **MyCalSet2A**
 - **Cal2B** at the Fixture B / DUT plane. Save to **MyCalSet2B**

Follow steps in [Create an S2P file...Step 6](#) above, except:

1. Create #1 S2P file:
 1. For CalSet1, choose **MyCalSet1** and select **CalPort=1**
 2. For CalSet2, choose **MyCalSet2A** and select **CalPort=1**

3. Save to **FixtureA.s2p**
2. Create #2 S2P file:
 1. For CalSet1, choose **MyCalSet1** and select **CalPort=2**
 2. For CalSet2, choose **MyCalSet2B** and select **CalPort=2**
 3. Save to **FixtureB.s2p**

Follow steps in [To De-embed the adapter](#) above, except:

1. For PNA Port 1, select **FixtureA.s2p**
 2. For PNA Port 2, select **FixtureB.s2p**
-

Last modified:

25-Mar-2010	Added support for version 2.1
23-Mar-2009	Fixed error in "De-embedding a fixture..." procedures
8-Dec-2008	Added support for version 2.0
3-Sep-2008	Removed legacy content
10-Mar-2008	Added Change Pivot dialog
13-Mar-2008	Moved FCA embed or de-embed
12-Feb-2008	New procedures
24-Jan-2008	Fixed error in procedures and added section
30-Nov-2007	Clarified and highlight order of calsets.
26-Feb-2007	Fixed " Note: Subsequent measurements...".
12-Sept-2006	Added link to programming commands

Swap Adapters Calibration Method

Although Swap Adapters calibration (also known as Swap Equal Adapters and Equal Length Adapters) method is NOT included in the PNA firmware, you can still perform this calibration.

Before we introduced the of the [Unknown Thru](#) method, the Swap Adapters method was often used as a quick alternative to the more tedious [adapter removal](#) method. In that case, you would be trading measurement accuracy for convenience. You might still want to perform Swap Adapters if you do NOT have calibration standards with the same connector type as your DUT. A procedure for this is shown below.

For any other reason, the Swap Adapters method is NOT recommended because the [Unknown Thru](#) method is more convenient AND more accurate.

There are many variations on the Swap Adapters calibration depending on the number of ports to be calibrated and whether the DUT is [insertable or non-insertable](#). However, the concepts are the same: you perform connect and measure the reflection standards (OPEN, SHORT, and LOAD) with one adapter in place, then swap for a different adapter for the Thru measurement. The adapters must have the same delay, loss, and impedance. The better the adapters match, the better measurement results. Measure the adapters with a calibrated PNA to be sure.

Procedure

The following is an example procedure showing how to perform a Swap Adapters 2-port calibration for a non-insertable DUT. The DUT has 2.92 mm connectors. You do NOT have 2.92 mm calibration standards, but you DO have 2.4 mm standards and adapters that have the same electrical properties as the 2.92 mm adapters.

Adapters A1 and A2 = test port to 2.4 mm adapters

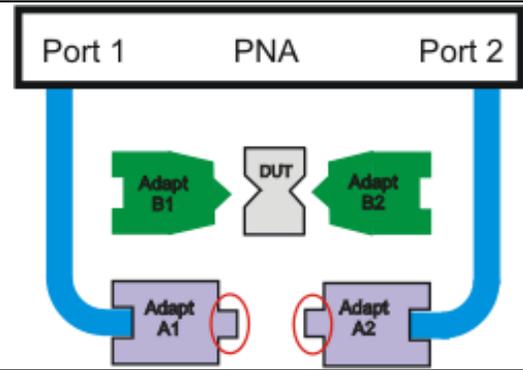
Adapters B1 and B2 = test port to 2.92 mm adapters

1. Start the Cal Wizard and select Guided (Smart) Cal. **Note:** The PNA will NOT prompt you to connect the adapters by name or when to swap the adapters.

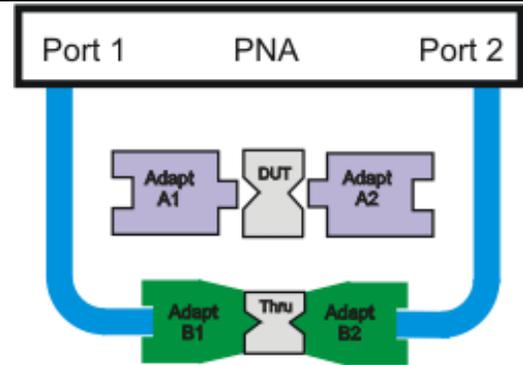
2. Specify the connector type and gender and Cal Kit of the adapter that you will be using (2.4 mm) - NOT the connector type of the DUT (2.92 mm). By specifying the connector gender, you are also specifying the Thru method (flush thru for insertable and Unknown Thru for non-insertable.) For example, when both DUT ports have female connectors, we will perform an Unknown Thru cal.

3. When prompted for reflection standards on port 1, connect the Open, Short, and Load standards to Adapter A1.

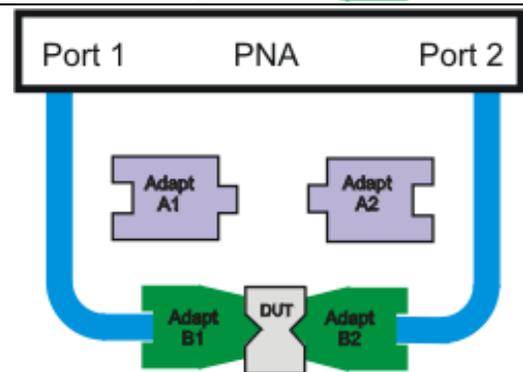
4. When prompted for reflection standards on port 2, connect the Open, Short, and Load standards to Adapter A2.



5. When prompted for a Thru connection, swap Adapter A1 and A2 for B1 and B2. Connect the Thru device. This could be any device that meets the requirements of the [Unknown Thru standard](#). In the case of a non-insertable DUT, connect B1 and B2.



6. Make DUT measurements with Adapters B1 and B2 in place.



Last Modified:

8-Dec-2010 MX New topic

Delta Match Calibration

A [TRL Cal](#), [QSOLT](#), or [Unknown Thru](#) Cal requires simultaneous, valid measurements of the reference receivers for the test ports being calibrated. This is not possible for both the **2-port AND 4-port** versions of the **N5231A, N5232A, and N5239A** PNA-L models.

This is also NOT possible with older N5230C 4-port models, nor for certain port pairs when an [external test set](#) is connected to ANY PNA model. With external test sets, some pairs of ports share the same receiver. In general these are the port pairs where one port is directly above the other. Therefore, TRL, QSOLT, and Unknown Thru are NOT possible in the context of a 2-port cal for any of those port pairs.

A Delta Match Calibration can be thought of as a software method used to overcome this hardware limitation. The Delta Match Calibration characterizes the [source match](#) and [load match](#) of the PNA test ports, and then calculates the differences, or "delta", of the two match terms. The delta term is combined with directivity and reflection tracking to compute a switch correction term. The switch correction error terms for each port are then used during subsequent TRL, QSOLT, or Unknown Thru calibrations.

There are several ways to acquire the Delta Match Calibration:

1. **From an existing User Cal Set** that meets the following Delta Match criteria: (Not allowed for use with [external test sets](#).)
 - Must have been performed using ECal or as a guided mechanical Cal (not Unguided).
 - Must have the same start frequency, stop frequency, and number of points as the channel being calibrated.
 - Must calibrate the ports that require the delta match terms.
2. **From a [Global Delta Match Calibration](#).**
3. **From a 'Self Delta Match'** when other portions of the calibration fully characterize all ports using SOLT with [Defined Thru](#) or [Flush Thru](#). For example, when calibrating all four ports of a PNA-L, perform a SOLT between ports 1 and 2, and also between ports 3 and 4, then Unknown Thru could be used between any combination of the remaining ports. This is allowed with an [external test set](#).

Which to use? A Self Delta Match Cal will always be used when possible. Otherwise, the Cal Wizard will use a GDM Cal when available unless you select [Choose Delta Match](#).

Global Delta Match (GDM) Cal

A GDM Cal is an "all-inclusive" calibration that can be applied whenever the delta match terms are

required.

Factory GDM Cal

Beginning with A.09.80, a factory-performed GDM calibration is installed on 2-port PNA-L models. This GDM Cal is identical to one that you would perform, except that it is likely to be more accurate than a GDM Cal that would be performed by most PNA-L customers. The factory GDM is used when a Self Delta Match Cal is not possible, just like a user-performed GDM.

- If you choose to perform a GDM Cal, it will overwrite the Factory GDM Cal.
- A backup copy of the factory GDM Cal is saved on the PNA-L at D:\calfiles\ Global Delta Match Calset.pcs.
- To restore the original file, copy it to the original location at: c:\users\public\network analyzer\UserCalSets\Global Delta Match Calset.pcs.
- If the original file is deleted, it will be restored automatically from the backup location.

User GDM Cal

When a GDM Cal is required, and your PNA-L model does NOT have a factory GDM, it must be performed at least once in order to use TRL, Unknown Thru, or QSOLT calibrations

- To attain the highest accuracy, the following settings are automatically used to perform a GDM Cal. When applied, it will likely be [interpolated](#).
 - Performed over the entire frequency range of the PNA.
 - Uses very dense data points, particularly at low frequencies.
 - Uses 100 Hz IF Bandwidth.
- The measurement conditions (cabling or adapters) of the subsequent calibration do NOT have to match the conditions under which the GDM Cal was performed. Because the GDM Cal characterizes the switch correction at each PNA port, those terms are NOT affected by differences in cabling or adapters.
- For highest accuracy when using an ECal module to perform the GDM Cal, the test ports of the ECal module should permit an [insertable connection](#) (for example 3.5mm male on one port and 3.5 mm female on the other). The frequency range of the ECal module must cover the entire frequency range. When using mechanical standards, the GDM Cal will force an insertable connection.
- Upon completion, the GDM Cal is stored as a special type of Cal Set and should be used ONLY as a Delta Match Cal. It provides Delta Match error terms, but does NOT provide all of the standard error correction terms.
- Experience has shown that a GDM Cal may require updating only once every year depending on environmental conditions and how heavily the instrument is used.

How to perform a GDM Cal

Note: 2-port PNA-L models have a Factory GDM Cal, which means it is NOT necessary to perform a GDM Cal. If you perform a GDM, the Factory GDM will be overwritten.

These selections will only be available if the PNA hardware requires a Delta Match Calibration.

Using front-panel HARDKEY [softkey] buttons

1. Press **CAL**
2. then **[Start Cal]**
3. then **[Global Delta Match]**

Using a mouse with PNA Menus

1. Click **Response**
2. then **Cal**
3. then **Start Cal**
4. then **Global Delta Match**

Programming Commands

Delta Match Calibration. Select DUT Connectors and Cal Kit dialog box help



- Only one Cal Kit is specified and necessary to perform a Delta Match Cal. However, ALL of the PNA test ports are calibrated in a Delta Match Cal.
- You must configure ALL test ports to terminate in the specified connector / gender using the necessary adapters. The errors from adapters are removed during calibration, but when using mechanical standards, the GDM Cal will force an insertable connection.
- For highest accuracy when using an ECal module to perform the GDM Cal, the test ports of the ECal module should permit an [insertable connection](#) (for example 3.5mm male on one port and 3.5 mm female on the other). The frequency range of the ECal module must cover the entire frequency range. When using mechanical standards, the GDM Cal will force an insertable connection. **If you select an ECal module that does NOT cover the entire frequency range of the PNA** your selection will change to a different Cal Kit.

Guided Calibration Steps dialog box help



Follow the prompts to connect standards to the calibration plane.

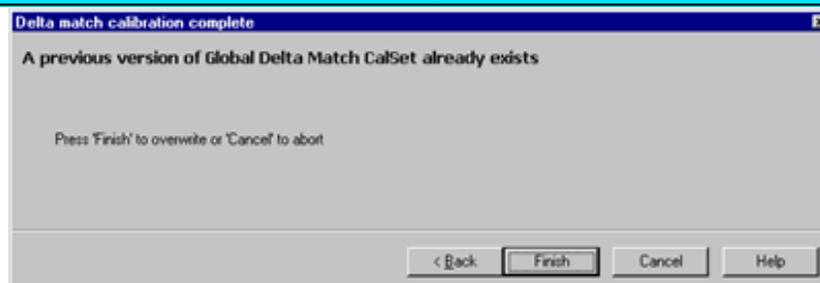
Then click **Measure**.

When all standards have been measured, click **Done** to complete the measurement steps.

THRU Connections

- ONLY 2 Thru connections are required for a 4-port Cal. This is less than the minimum number of Thrus of a standard 4-port Cal.
- GDM Cal is always performed using a **Flush Thru**, a **Known Thru**, or an **insertable ECal module**. You can NOT use an Unknown Thru in the calibration process. For highest accuracy, perform GDM Cal using an insertable ECal module and select Flush-thru as the Calibration Thru method. [Learn about Calibration Thru methods.](#)

Delta Match Calibration Complete dialog box help



Click **Finish** to store the GDM Calibration as a special type of Cal Set.

By default, it will be used when a Delta Match Calibration is required.

It can ONLY be used as a Delta Match Cal. It does NOT provide all of the standard error correction terms.

Last modified:

7-Jan-2013	Expanded on external testsets (BH)
25-Oct-2012	Added Factory GDM
18-Oct-2011	Added clarification of conditions and timing
3-Sep-2008	Removed legacy content
9-Nov-2007	Edits for requirements
23-Feb-2007	Modified requirements for multiport
9/12/06	Added link to programming commands

Calibration Overview

The following is discussed in this topic:

- [What Is Measurement Calibration?](#)
- [Why Is Calibration Necessary?](#)
- [Conditions Where Calibration Is Suggested](#)
- [What Is ECal?](#)

See other Calibration Topics

What Is Measurement Calibration?

Calibration removes one or more of the systematic errors using an equation called an error model. Measurement of high quality standards (for example, a short, open, load, and thru) allows the analyzer to solve for the error terms in the error model. See [Measurement Errors](#).

You can choose from different calibration types, depending on the measurement you are making and the level of accuracy you need for the measurement. See [Select a Calibration Type](#).

The accuracy of the calibrated measurements is dependent on the quality of the standards in the calibration kit and how accurately the standards are modeled (defined) in the calibration kit definition file. The calibration-kit definition file is stored in the analyzer. In order to make accurate measurements, the calibration-kit definition must match the actual calibration kit used. To learn more, see [Accurate Calibrations](#).

Calibration Wizard provides the different calibration methods used in the PNA. See [Calibration Wizard](#).

There are quick checks you can do to ensure your measurement calibration is accurate. To learn more see [Validity of a Measurement Calibration](#)

If you make your own custom-built calibration standards (for example, during in-fixture measurements), then you must characterize the calibration standards and enter the definitions into a user modified calibration-kit file. For more information on modifying calibration kit files, see [Calibration Standards](#).

Note: [Instrument Calibration](#) is ensuring the analyzer hardware is performing as specified. This is not the same as measurement calibration.

Why Is Calibration Necessary?

It is impossible to make perfect hardware that would not need any form of **error correction**. Even making the hardware good enough to eliminate the need for error correction for most devices would be extremely expensive.

The accuracy of network analysis is greatly influenced by factors external to the network analyzer. Components of the measurement setup, such as interconnecting cables and adapters, introduce variations in magnitude and **phase** that can mask the actual response of the device under test.

The best balance is to make the hardware as good as practically possible, balancing performance and cost. Calibration is then a very useful tool to improve measurement accuracy.

Conditions Where Calibration Is Suggested

Generally, you should calibrate for making a measurement under the following circumstances:

- You want the best accuracy possible.
- You are adapting to a different connector type or impedance.
- You are connecting a cable between the test device and an analyzer test port.
- You are measuring across a wide frequency span or an electrically long device.
- You are connecting an attenuator or other such device on the input or output of the test device.

If your test setup meets any of the conditions above, the following system characteristics may be affected:

- Amplitude at device input
- Frequency response accuracy
- Directivity
- Crosstalk (isolation)
- Source match
- Load match

What Is ECal

ECal is a complete solid-state calibration solution. It makes one port (Reflection), full two and three-port calibrations fast and easy. See [Using ECal](#).

- It is less prone to operator error.

- The various standards (located inside the calibration module) never wear out because they are switched with PIN-diode or FET switches.
- The calibration modules are characterized using a TRL-calibrated network analyzer.
- ECal is not as accurate as a good TRL calibration.

For information about ordering ECal modules, see [Analyzer Accessories](#) or contact your [Agilent Support Representative](#)

Measurement Errors

You can improve accuracy by knowing how errors occur and how to correct for them. This topic discusses the sources of measurement error and how to monitor error terms.

- [Drift Errors](#)
- [Random Errors](#)
- [Systematic Errors](#)
 - [3-Port Error Terms](#)
 - [4-Port Error Terms](#)
- [Monitoring Error Terms](#)

[See other Calibration Topics](#)

Drift Errors

Drift errors are due to the instrument or test-system performance changing after a calibration has been done.

Drift errors are primarily caused by thermal expansion characteristics of interconnecting cables within the test set and conversion stability of the microwave frequency converter and can be removed by re-calibrating.

The time frame over which a calibration remains accurate is dependent on the rate of drift that the test system undergoes in your test environment.

Providing a stable ambient temperature usually minimizes drift. For more information, see [Measurement Stability](#).

Random Errors

Random errors are not predictable and cannot be removed through error correction. However, there are things that can be done to minimize their impact on measurement accuracy. The following explains the three main sources of random errors.

Instrument Noise Errors

Noise is unwanted electrical disturbances generated in the components of the analyzer.

These disturbances include:

- Low level noise due to the broadband noise floor of the receiver.
- High level noise or jitter of the trace data due to the noise floor and the phase noise of the LO source inside the test set.

You can reduce noise errors by doing one or more of the following:

- Increase the [source power](#) to the device being measured - ONLY reduces low-level noise.
- [Narrow the IF bandwidth](#).
- Apply several measurement [sweep averages](#).

Switch Repeatability Errors

Mechanical RF switches are used in the analyzer to switch the source attenuator settings.

Sometimes when mechanical RF switches are activated, the contacts close differently from when they were previously activated. When this occurs, it can adversely affect the accuracy of a measurement.

You can reduce the effects of switch repeatability errors by avoiding switching attenuator settings during a critical measurement.

Connector Repeatability Errors

Connector wear causes changes in electrical performance. You can reduce connector repeatability errors by practicing good connector care methods. See [Connector Care](#).

Systematic Errors

Systematic errors are caused by imperfections in the analyzer and test setup.

- They are repeatable (and therefore predictable), and are assumed to be time invariant.
- They can be characterized during the calibration process and mathematically reduced during measurements.
- They are never completely removed. There are always some residual errors due to limitations in the calibration process. The residual (after measurement calibration) systematic errors result from:
 - imperfections in the calibration standards
 - connector interface
 - interconnecting cables

- instrumentation

Reflection measurements generate the following three systematic errors:

- [Directivity](#)
- [Source Match](#)
- [Frequency Response Reflection Tracking](#)

Transmission measurements generate the following three systematic errors:

- [Isolation](#)
- [Load Match](#)
- [Frequency Response Transmission Tracking](#)

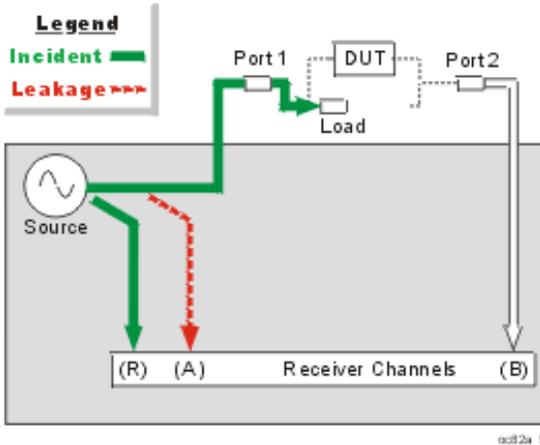
Notes about the following Systematic Error descriptions:

- The figures for the following six systematic errors show the relevant hardware configured for a forward measurement. For reverse measurements, internal switching in the analyzer makes Port 2 the source and Port 1 the receiver. 'A' becomes the transmitted receiver, 'B' becomes the reflected receiver, and 'R2' becomes the reference receiver. These six systematic errors, times two directions, results in 12 systematic errors for a two port device.
- For simplicity, it may be stated that ONE standard is used to determine each systematic error. In reality, ALL standards are used to determine ALL of the systematic errors.
- The following describes an SOLT calibration. This does not apply to TRL, or other types of calibration.

Directivity Error

All network analyzers make reflection measurements using directional couplers or bridges.

With an ideal coupler, only the reflected signal from the DUT appears at the 'A' receiver. In reality, a small amount of incident signal leaks through the forward path of the coupler and into the 'A' receiver. This leakage path, and any other path that allows energy to arrive at the 'A' receiver without reflecting off the DUT, contributes to directivity error.



How the Analyzer Measures and Reduces Directivity Error.

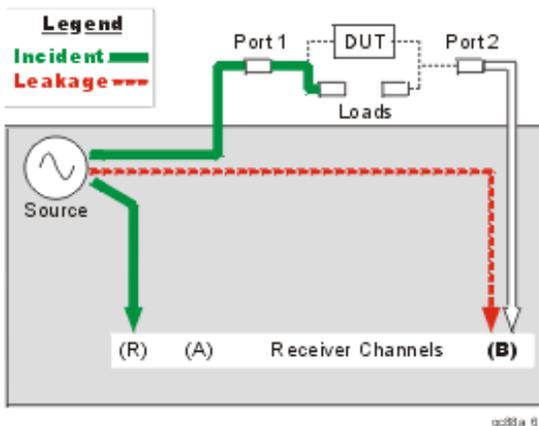
1. During calibration, a load standard is connected to Port 1. We assume no reflections from the **load**.
2. The signal measured at the 'A' receiver results from the incident signal leakage through the coupler and other paths.
3. Directivity error is mathematically removed from subsequent reflection measurements.

Isolation Error

Ideally, only signal transmitted through the DUT is measured at the 'B' receiver.

In reality, a small amount of signal leaks into the 'B' receiver through various paths in the analyzer.

The signal leakage, also known as crosstalk, is isolation error which can be characterized and reduced by the analyzer.



How the Analyzer Measures and Reduces Isolation Error

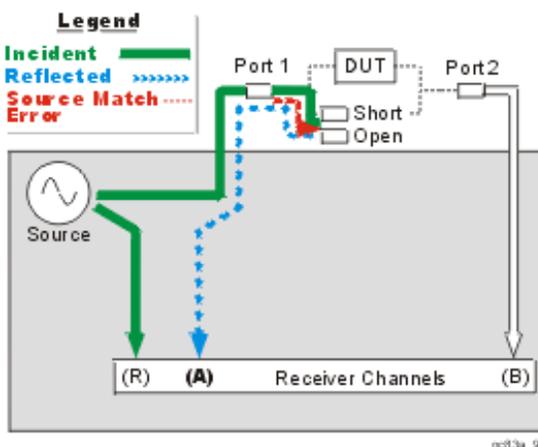
1. During calibration, load standards are connected to both Port 1 and Port 2.
2. The signal measured at the 'B' receiver is leakage through various paths in the analyzer.
3. This isolation error is mathematically removed from subsequent transmission measurements.

Source Match Error

Ideally in reflection measurements, all of the signal that is reflected off of the DUT is measured at the 'A' receiver.

In reality, some of the signal reflects off the DUT, and multiple internal reflections occur between the analyzer and the DUT. These reflections combine with the incident signal and are measured at the 'A' receiver, but not at the 'R' receiver.

This measurement error is called source match error which can be characterized and reduced by the analyzer.



How the Analyzer Measures and Reduces Source Match Error

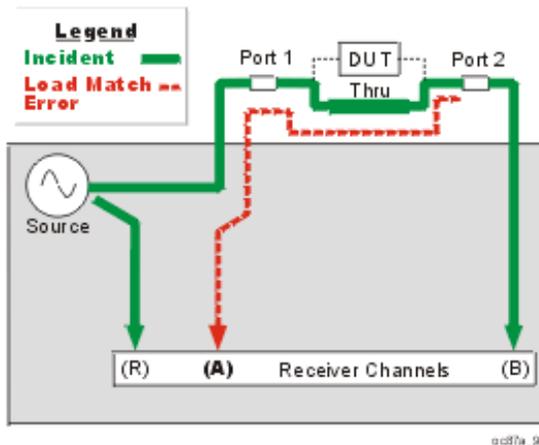
1. During calibration, all reflection standards are connected to Port 1. Known reflections from the standards are measured at the 'A' receiver.
2. Complex math is used to calculate source match error.
3. Source match error is mathematically removed from subsequent reflection and transmission measurements.

Load Match Error

Ideally in transmission measurements, an incident signal is transmitted through the DUT and is measured at the 'B' receiver.

In reality, some of the signal is reflected off of Port 2 and other components and is not measured at the 'B' receiver.

This measurement error is called load match error which can be characterized and reduced by the analyzer.



How the Analyzer Measures and Reduces Load Match Error

1. The Port 1 and Port 2 test connectors are mated together for a perfect zero-length thru connection. If this is not possible, a [characterized thru adapter](#) is inserted. This allows a known amount of incident signal at Port 2.
2. The signal measured at the 'A' receiver is reflection signal off of Port 2
3. The resulting load match error is mathematically removed from subsequent transmission and reflection measurements.

Frequency Response Reflection Tracking Error

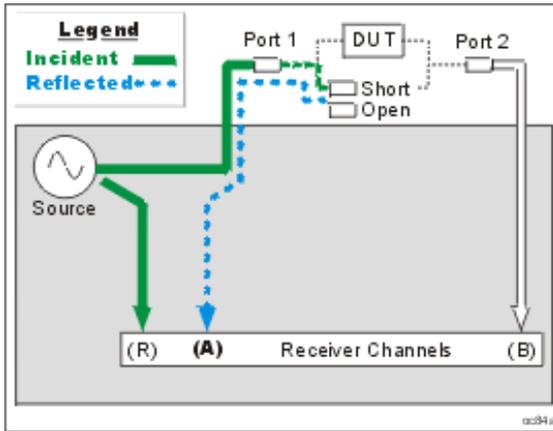
Reflection measurements are made by comparing signal at the 'A' receiver to signal at the 'R1' receiver. This is called a ratio measurement or "A over R1" (A/R1).

For ideal reflection measurements, the frequency response of the 'A' and 'R1' receivers would be identical.

In reality, they are not, causing a frequency response reflection tracking error. This is the vector sum of all test variations in which magnitude and phase change as a function of frequency. This includes variations contributed by:

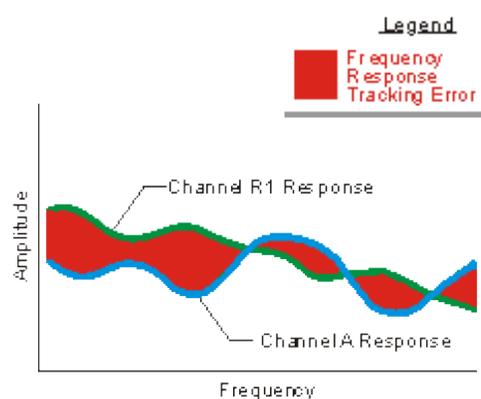
- signal-separation devices
- test cables
- adapters
- variations between the reference and test signal paths

Frequency response reflection tracking error can be characterized and reduced by the analyzer.



How the Analyzer Measures and Reduces Frequency Response Reflection Tracking Error.

1. During calibration, all reflection standards are used to determine reflection tracking.
2. The average 'A' receiver response is compared with the 'R1' receiver response.
3. Complex math is used to calculate Frequency Response Reflection Tracking Error (see the following diagram). This frequency response reflection tracking error is mathematically removed from subsequent DUT measurements.



Note: In reflection response calibrations, only a single calibration standard is measured (open or short) and thus only its contribution to the **error correction** is used.

Frequency Response Transmission Tracking Error

Transmission measurements are made by comparing signal at the 'B' receiver to signal at the 'R1' receiver. This is called a ratio measurement or "B over R1" (B/R1).

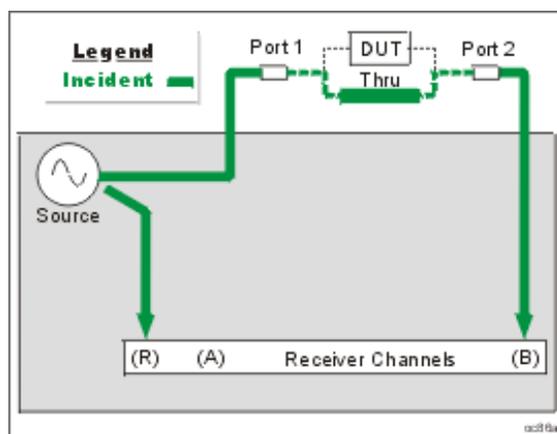
For ideal transmission measurements, the frequency response of the 'B' and 'R1' receivers would be identical.

In reality, they are not, causing a frequency response transmission tracking error. This is the vector sum

of all test variations in which magnitude and phase change as a function of frequency. This includes variations contributed by:

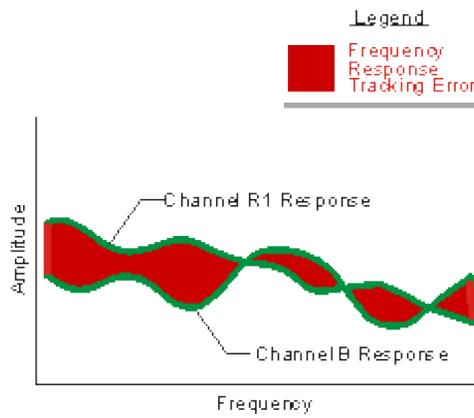
- signal-separation devices
- test cables
- adapters
- variations between the reference and test signal paths

Frequency response transmission tracking error can be characterized and reduced by the analyzer.



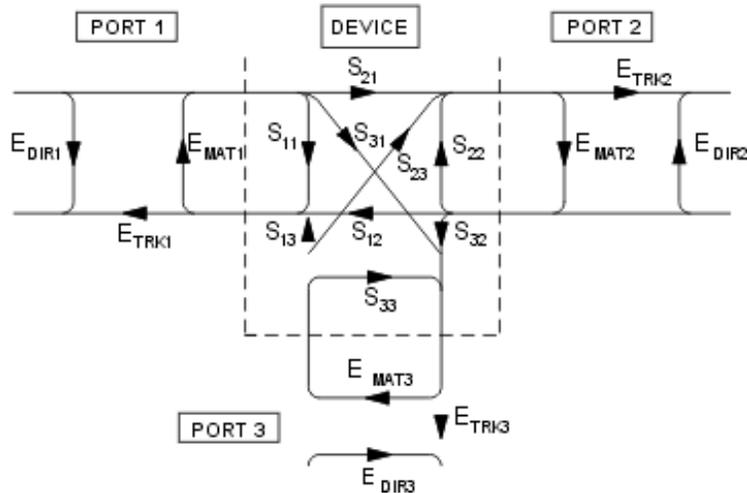
How the Analyzer Measures and Reduces Frequency Response Transmission Tracking Error.

1. During calibration, the Port 1 and Port 2 test connectors are mated together for a perfect zero-length thru connection. If this is not possible, a [characterized thru adapter](#) is inserted. This allows a known amount of incident signal to reach Port 2.
2. Measurements are made at the 'B' and 'R1' receivers.
3. Complex math is used to calculate Frequency Response Transmission Tracking Error (see the following diagram). This frequency response transmission tracking error is mathematically removed from subsequent DUT measurements.



3-Port Error Terms

The following flow diagram displays the 3-port error term model:



where:

E = error term

DIR = Directivity

MAT = Forward Source Match and Reverse Load Match

TRK = Forward Reflection Tracking and Reverse Transmission Tracking

4-Port error terms

A full 4-port calibration requires the following terms:

[Learn about the port numbering convention](#) for error terms.

		Source Port			
		1	2	3	4
R e c e i v e	1	DIR 1,1 RTRK 1,1 SRM 1,1	LDM 1,2 TTRK 1,2 XTLK 1,2	LDM 1,3 TTRK 1,3 XTLK 1,3	LDM 1,4 TTRK 1,4 XTLK 1,4
	2	LDM 2,1 TTRK 2,1 XTLK 2,1	DIR 2,2 RTRK 2,2 SRM 2,2	LDM 2,3 TTRK 2,3 XTLK 2,3	LDM 2,4 TTRK 2,4 XTLK 2,4
	3	LDM 3,1 TTRK 3,1 XTLK 3,1	LDM 3,2 TTRK 3,2 XTLK 3,2	DIR 3,3 RTRK 3,3 SRM 3,3	LDM 3,4 TTRK 3,4 XTLK 3,4
	4	LDM 4,1 TTRK 4,1 XTLK 4,1	LDM 4,2 TTRK 4,2 XTLK 4,2	LDM 4,3 TTRK 4,3 XTLK 4,3	DIR 4,4 RTRK 4,4 SRM 4,4

Reflection terms

- DIR: Directivity

- RTRK: Reflection Tracking
- SRM: Source Match

Transmission terms

- LDM: Load Match
- TTRK: Transmission Tracking
- XTLK: Cross Talk

How can we measure only 3 THRU connections?

On a 4-port PNA, a full 4-port cal can be performed while measuring only 3 THRU connections. Measuring more than 3 THRU connections will give higher accuracy.

By measuring all of the reflection terms, and 3 transmission THRU connections, there is adequate information available to calculate the remaining transmission terms. The following is a high level explanation of the concept. The actual calculations are much more complex.

To simplify, let's substitute letters (A,B,C,D) for port numbers from the diagram above so that they can be combined without confusion. Also for simplicity, let's assume that the source match and directivity errors are zero.

	A	B	C	D
A	AA	AB	AC	AD
B	BA	BB	BC	BD
C	CA	CB	CC	CD
D	DA	DB	DC	DD

- The reflection errors are all measured (AA, BB, CC, DD).
- Lets assume we measure a THRU between ports AB, AC, AD. The reverse direction for these THRU's are also measured at the same time (BA, CA, DA).
- The terms left to calculate are BC, CB, BD, DB, CD, DC.

The following shows how the BC term is calculated from BA and AC:

$$\frac{BA * AC}{AA} = \frac{B * \cancel{AA} * C}{\cancel{AA}} = BC$$

Similarly:

- CB is calculated from CA and AB
- BD is calculated from BA and AD
- DB is calculated from AB and DA
- CD is calculated from CA and AD
- DC is calculated from DA and AC

Monitoring Error Terms using Cal Set Viewer

You can use **Cal Set Viewer** to monitor the measured data and the calculated error term. This will help to determine the health of your PNA and the accuracy of your measurements.

By printing or saving the error terms, you can periodically compare current error terms with previously recorded error terms that have been generated by the same PNA, measurement setup, and calibration kit. If previously generated values are not available, refer to Typical Error Term Data in Appendix A, "Error Terms", of the Service Guide.

Note: The service guide for your PNA is available at <http://www.agilent.com/find/pna>
It is also on the CDROM that was shipped with your PNA.

- A stable system should generate repeatable error terms over about six months.
- A sudden shift in error terms over the same frequency range, power, and receiver settings, may indicate the need for troubleshooting system components. For information on troubleshooting error terms, see Appendix A , "Error Terms", of the Service Guide.
- A subtle, long-term shift in error terms often reflects drift or connector and cable wear. The cure is often as simple as cleaning and gauging connectors or inspecting cables.

Viewing Cal Set Data

- Existing measurement traces are unaffected by the Cal Set Viewer.
- The Cal Set data trace is presented in the highest unused channel number (usually 32) in the active window.
- The Cal Set data trace is labeled as S11 in the status bar regardless of the type of error term or standard.
- Only one Cal Set error term or standard data can be viewed at a time. However, a data trace can be stored into memory and then compared to other data traces.

How to access Cal Set Viewer

Using front-panel HARDKEY [softkey] buttons

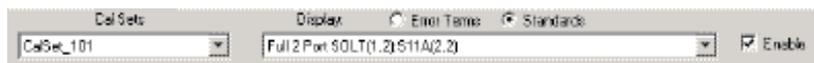
1. Press **CAL**
2. then **[Manage Cals]**
3. then **[Cal Set Viewer]**

Using a mouse with PNA Menus

1. Click **Response**
2. then **Cal**
3. then **Manage Cals**
4. then **Cal Set Viewer**

Programming Commands

How to use Cal Set Viewer



1. Use the down arrow to select a Cal Set. Then click either:
 - **Error Terms** - calculated data.
 - **Standards** - the raw measurement data of the Standard. **ONLY** available with Unguided Cal (not ECal or Guided Cal).
2. Use the down arrow to select an error term or standard to view.
3. Select the **Enable** check box to view the data on the PNA screen.

Port numbering convention for error terms is the same as for S-Parameters:

E Term (Receiver, Source) with the following exceptions:

- Load Match (2,1) - The match of port 2 which is measured by making an S11 measurement.
 - Load Match (1,2) - The match of port 1 which is measured by making an S22 measurement.
 - Transmission Tracking (2,1) - The port 2 receiver relative to the port 1 reference. (source=port 1).
 - Transmission Tracking (1,2) - The port 1 receiver relative to the port 2 reference. (source=port 2).
 - And so forth for multipoint calibrations.
-

Last modified:

3-Sep-2008 Removed legacy content

Accurate Measurement Calibrations

Calibration accuracy is affected by the type of calibration, quality of the calibration standards, and the care with which the calibration is performed. This section provides additional information about how to make accurate calibrations.

- [Measurement Reference Plane](#)
- [Effects of Using Wrong Calibration Standards](#)
- [Data-based versus Polynomial Calibration Kits](#)
- [Accuracy Level of Interpolated Measurement](#)
- [Effects of Power Level](#)
- [Using Port Extensions](#)
- [Isolation Portion of 2-Port Calibration](#)
- [Choosing a Thru Method](#)

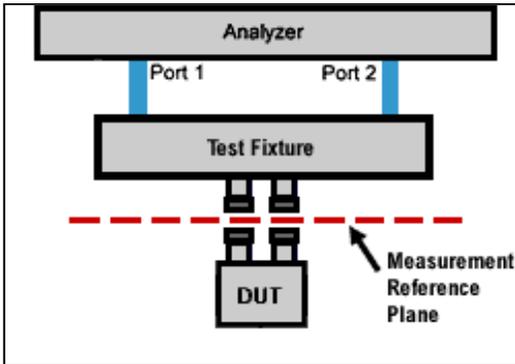
Learn how to [determine the validity of your calibration](#).

[See other Calibration Topics](#)

Measurement Reference Plane

Most measurement setups will NOT allow you to connect a device under test (DUT) directly to the PNA front panel test ports. More likely, you would connect your device to test fixtures, adapters, or cables that are connected to the PNA.

A calibration takes place at the points where calibration standards are connected during the calibration process. This is called the measurement reference plane (see graphic). For the highest measurement accuracy, make the calibration reference plane the place where your DUT is connected. When this occurs, the errors associated with the test setup (cables, test fixtures, and adapters used between the analyzer ports and the reference plane) are measured and removed in the calibration process.



Effects of Using Wrong Calibration Standards

Normally, a calibration is performed using a calibration kit that contains standards with connectors of the same type and sex as your DUT.

However, your calibration kit may not always have the same connector type and gender as your device. For example, suppose your device has 3.5mm connectors, but you have a Type-N calibration kit. If you use an adapter to connect the Type-N standards to the 3.5mm test port, then the adapter becomes part of the calibration and NOT part of the test setup. This will result in significant errors in your reflection measurements.

Data-based versus Polynomial Calibration Kits

The [Select DUT Connectors and Cal Kits](#) dialog box offers a data-based model and a polynomial model for the newest high-frequency cal kits. See PNA Accessories. The data-based models provide higher accuracy for describing calibration standards than the polynomial models. It is RECOMMENDED that the data-based model be used if the most accurate results are desired.

	Data-Based Model	Polynomial Model
How accurate is the model?	Provides highest calibration accuracy. Eliminates the errors that can be the result of polynomial model approximations.	Provides high calibration accuracy.
How does the model define calibration standards?	Uses S-Parameter measurements.	Uses traditional four-term polynomial calibration standard modeling parameters.
How do I manually edit the definitions of the calibration standards when using the model?	Use the Advanced Modify Cal Kit function.	Use the Advanced Modify Cal Kit function.

How do I use the Calibration Wizard with the model?	Use only the SmartCal (Guided) Calibration method.	Use the SmartCal (Guided) or the Unguided Mechanical Calibration methods.
--	--	---

Learn about the [“Expanded Math”](#) feature.

Effects of Power Level

To attain the most accurate error correction, do NOT change the power level after a calibration is performed. However, when changing power within the same attenuator range at which the measurement calibration was performed, S-parameter measurements can be made with only a small degradation of accuracy. If a different attenuator range is selected, the accuracy of error correction is further degraded.

To check the accuracy of a calibration, see [Validity of a Calibration](#).

Using Port Extensions

Use the port extensions feature after calibration to compensate for phase shift of an extended measurement reference plane due to additions such as cables, adapters, or fixtures.

Port extensions is the simplest method to compensate for phase shift, mismatch, and loss of the path between the calibration reference plane and the DUT.

Learn how to apply [port extensions](#).

Learn about [characterizing a test fixture](#).

Isolation Portion of 2-Port Calibration

The isolation portion of a calibration corrects for crosstalk, the signal leakage between test ports when no device is present. When performing an UNGUIDED 2-port calibration, you have the option of omitting the isolation portion of the calibration.

Note: Isolation can be performed on a Smart (Guided) Calibration ONLY remotely using SCPI or COM.

The uncorrected isolation between the test ports of the PNA is exceptional (typically >100dB). Therefore, you should only perform the Isolation portion of a 2-port calibration when you require isolation that is better than 100dB. Perform an isolation calibration when you are testing a device with high insertion loss, such as some filter stopbands or a switch in the open position.

The isolation calibration can add noise to the error model when the measurement is very close to the noise floor of the analyzer. To improve measurement accuracy, set a narrow IF Bandwidth.

How to perform an isolation calibration

Isolation is measured when the Load standards are connected to the PNA test ports. For best accuracy, connect Load standards to BOTH test ports each time you are prompted to connect a load standard. If two Loads are not available, connect the untested PNA port to any device that will present a good match.

Choosing a Thru Method

When calibrating for a non-insertable device, you must choose a method to calibrate for the THRU error terms. This can have a significant effect on measurement accuracy. Learn more about [choosing a thru method](#).

Last Modified:

18-Feb-2011 Removed popups and added remotely to Isolation note.

Calibration Thru Methods

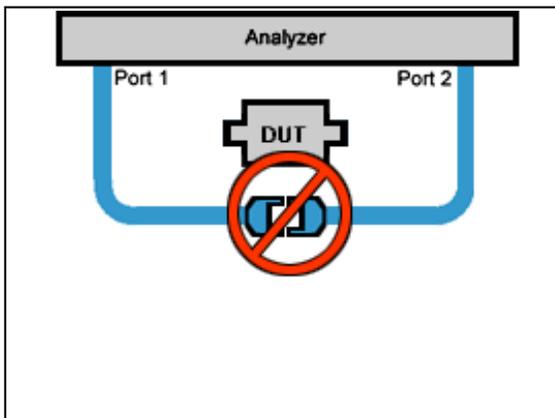
- [What is a Non-Insertable Device](#)
- [Choosing a Thru Method](#)
- [Flush Thru](#)
- [Adapter Removal](#)
- [Swap Adapters](#) (separate topic)
- [Defined Thru](#)
- [Unknown Thru](#)
- [ECal Thru Method Choices](#)

Other Cal Topics

What is a Non-Insertable Device

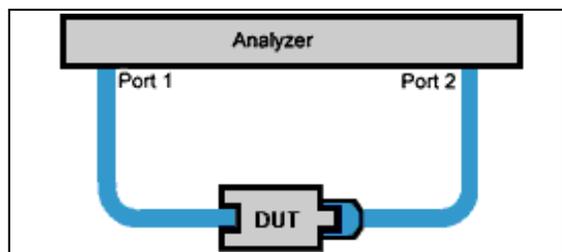
To understand the Thru method choices, you must first understand what is meant by "Non-Insertable device". These definitions also apply to ECal modules. Substitute "ECal module" for "device". Then see [ECal Thru Method Choices](#).

A **non-insertable device** is one whose connectors could NOT mate together. They either do not have the same type of connector or they have the same gender. This also means that the test port cables would not mate together, as in the following diagram.



An **insertable** device is one whose connectors could mate together. They have the same type of

connector and opposite, or no, gender. This also means that the test port cables would mate together, as in the following diagram.



Choosing a Thru Method of Calibration

The Thru method is selected from the Cal Wizard. Select the **Modify** checkbox in the [Select DUT Connectors and Cal Kits](#) dialog box.

Notes:

For ECal, the following choices have different meanings. See [THRU methods for ECal](#).

For 4-port calibration, see [How can we measure only 3 THRU connections?](#)

Choice for Insertable Devices: FLUSH Thru (also known as Zero-length Thru)

When calibrating for an insertable device, the test ports at your measurement reference plane connect directly together. This is called a zero-length THRU, or Flush THRU meaning that the THRU standard has zero-length: no delay, no loss, no capacitance, and no inductance. Your calibration kit may not have a physical THRU standard because it is assumed you have an insertable device and will be using a zero-length THRU.

Choices for Non-Insertable Devices

The following methods calibrate for a non-insertable device:

- [Adapter Removal](#) Accurate, but least convenient.
- [Defined Thru](#)
- [Unknown Thru Cal](#) Preferred method.

Adapter Removal Calibration

This method is potentially very accurate. However, it requires many connections which increases the chances of inaccurate data.

Two full 2-port calibrations are performed: one with the adapter connected at port 1, and the other with the adapter connected to port 2. The result of the two calibrations is a single full 2-port calibration that includes accurate characterization and removal of the mismatch caused by the adapter.

Performing an Adapter Removal Cal requires:

- a THRU adapter with connectors that match those on the DUT.
- calibration standards for both DUT connectors.

To select Adapter Removal during a SmartCal, select the **Modify** checkbox in the [Select DUT Connectors and Cal Kits](#) dialog box. The Cal Wizard will guide you through the steps.

Learn how to perform an [Adapter Removal Cal using ECal](#).

Defined Thru (also known as **Known Thru**, **Cal Kit Thru**, **ECal Thru**, **Characterized Thru**)

Defined Thru uses the THRU definition that is stored in the Cal Kit file or ECal module. The THRU standard may have worn over time, making it not as accurate as when it was new. Defined Thru is usually more accurate than Adapter Removal, but not as accurate as [Unknown Thru](#) method.

Notes

- If performing an ECal, this is the THRU standard in the ECal Module.
- If Defined Thru appears as a potential THRU method in the [SmartCal Wizard](#), this means that there is a defined THRU standard in the selected Cal Kit. This could be a [Zero-length Thru](#). The SmartCal Wizard will prompt you to connect the required standard when appropriate.

To define a THRU standard in a Cal Kit (not ECal module):

1. From the PNA Menu, click Calibration, [Advanced Modify Cal Kits](#).
2. Select the Cal Kit
3. Click Edit Kit
4. Click Add
5. Select THRU
6. Complete the dialog box.

The next time you perform a Guided Cal, this Defined THRU standard will be available if the DUT connector types match the THRU standard.

Unknown Thru Cal

Unknown Thru Cal is the **preferred** THRU method of calibrating the PNA to measure a non-insertable device.

The Unknown Thru calibration is also known as **Short-Open-Load-Reciprocal Thru** (SOLR) calibration.

- Very easy to perform.
- Better accuracy than [Defined Thru](#) and usually better than [Adapter Removal](#).
- Does not rely on existing standard definitions that may no longer be accurate.
- Causes minimal cable movement if the THRU standard has the same footprint as the DUT. In fact, the DUT can often BE the THRU standard.
- NOT recommended when there is 40 dB or more of combined loss in the Unknown Thru and calibration path. This would NOT allow enough signal to accurately measure at the receiver.

About the Unknown Thru Process

SmartCal guides you through the process. Although the following process describes ports 1 and 2, Unknown Thru can be performed on any two ports when using a multiport PNA.

1. Perform 1-port cal on port 1.
2. Perform 1-port cal on port 2.
3. Connect Unknown Thru between ports 1 and 2.
4. Measure Unknown Thru.
5. [Confirm Estimated Delay](#). This estimate may be wrong if there are too few frequency points over the given frequency span. You can measure the delay value independently and enter that value in the dialog box.

The Unknown Thru Standard

- Can have up to 40 dB of combined loss in the Unknown Thru and calibration path.
- Must be reciprocal: $S_{21}=S_{12}$.
- Must know the phase response to within 1/4 wavelength (see step 5 above).
- Can be the DUT if it meets these conditions.

Unknown Thru Limitations

- Unknown Thru is NOT supported during a TRL calibration.

- Beginning with PNA code release 5.25, Unknown Thru CAN be performed using a 4-port PNA-L that does NOT have a [reference receiver for each test port](#). However, a [Delta Match Calibration](#) is usually required before the Unknown Thru is measured.
- Unknown Thru is NOT supported on E8801A, E8802A, and E8803A.

ECal Thru Method Choices

When the ECal module connectors exactly match the DUT connectors, choose from the following THRU methods:

ECal Thru as Unknown Thru [Learn more about Unknown Thru.](#)

- Measures the THRU state of the ECal module as an Unknown Thru.
- The default method when the ECal module connectors match the DUT.
- Very accurate and easy.
- May require a [Delta Match Cal.](#)

Flush Thru (zero-length Thru) [Learn more about Flush Thru](#)

- Requires an insertable ECal module / DUT.
- Remove the ECal module and connect the two reference planes directly together for a zero-length thru.
- Accurate, but not as easy as 'ECal Thru as Unknown Thru'.

ECal (Defined Thru)

- Measures the THRU state of the ECal module.
- Very easy, but not as accurate as 'ECal Thru as Unknown Thru'

Unknown Thru

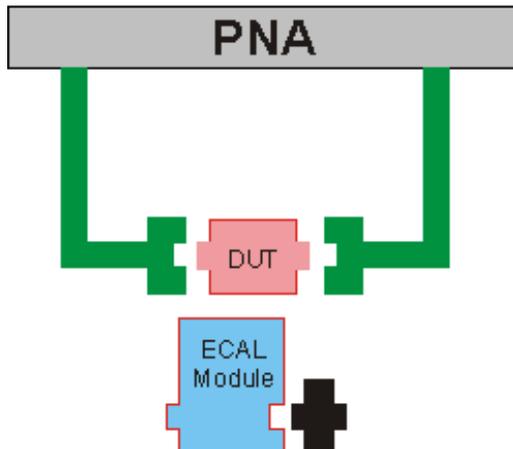
- Remove the ECal module.
- Then connect a Thru adapter to be measured as Unknown Thru.
- May require a [Delta Match Cal.](#)

When the ECal module connectors do NOT exactly match the DUT connectors, choose from the following two methods:

Adapter Removal

- Can be used with ECal when your DUT is [NON-insertable](#). However, the ECal module MUST be insertable, and the adapter connectors must exactly match the connectors of the DUT as in the following diagram.

Note: With PNA release 4.8, adapter removal now performs 2-port measurements on both sides of the adapter. It previously performed 2-port measurements on one side and 1-port measurements on the other. This improves the accuracy of the adapter removal calibration.



ECal User Characterization

In cases when adapter removal cannot be performed, ECal [User Characterization](#) is ALWAYS possible if you have the right adapters. A User Characterization is performed once and stored in the ECal module. However, accuracy is compromised every time you remove, then reconnect, the adapter with the ECal module.

Last Modified:

25-Oct-2012	Edited Unknown Thru std.
15-Dec-2010	Added link to swap equal adapter
20-Feb-2008	Added bullet for default ECal method

Validity of a Calibration

This section helps you determine if your calibration is valid and how the analyzer displays correction level information for your measurement.

- [Frequency Response of Calibration Standards](#)
- [Validating a Calibration](#)
- [Quick Check](#)
- [ECal Confidence Check](#)
- [Verification Kit](#)

See other Calibration Topics

Frequency Response of Calibration Standards

In order for the response of a calibration standard to show as a dot on the [smith chart display format](#), it must have no phase delay with respect to frequency. The only standards that exhibit such "perfect" response are the following:

- 7-mm short (with no offset)
- Type-N male short (with no offset)

There are two reasons why other types of calibration standards show phase delay after calibration:

1. The reference plane of the standard is electrically offset from the mating plane of the test port. Such devices exhibit the properties of a small length of transmission line, including a certain amount of phase shift.
2. The standard is an open termination, which by definition exhibits a certain amount of fringe capacitance and therefore phase shift. Open terminations which are offset from the mating plane will exhibit a phase shift due to the offset in addition to the phase shift caused by the fringe capacitance.

The most important point to remember is that all standards are measured in order to remove [systematic errors](#) from subsequent device measurements. As a result, if calibration standards with delay and fringe capacitance are measured as a device after a calibration, they will NOT appear to be "perfect". This is an indication that your analyzer **is calibrated accurately and working properly**.

Validating a Calibration

At the completion of a calibration or selection of a stored Cal Set, validation can accomplish the following:

Improve Measurement Accuracy – Once a measurement calibration has been performed, its performance should be checked before making device measurements. There are several sources of error that can invalidate a calibration: bad cables, dirty or worn calibration standards that no longer behave like the modeled standards, and operator error.

Verify Accuracy of Interpolation – You should validate the calibration if you are testing a device and the measurements are uncertain because of interpolation. For more information see [Interpolation Accuracy](#).

Verify Accuracy of Cal Standards – To check accuracy, a device with a known magnitude and phase response should be measured.

Quick Check

For this test, all you need are a few calibration standards. The device used should not be one of the calibration standards; a measurement of one of these standards is merely a measure of repeatability.

The following reflection and transmission Quick Check tests can be applied to all test ports.

To verify reflection measurements, perform the following steps:

1. Connect either an OPEN or SHORT standard to port 1. The magnitude of S11 should be close to 0 dB (within a few tenths of a dB).
2. Connect a load calibration standard to port 1. The magnitude of S11 should be less than the specified calibrated directivity of the analyzer (typically less than -30 dB).

To verify transmission measurements:

1. Connect a THRU cable (or known device representative of your measurement) from port 1 to port 2. Verify the loss characteristics are equivalent to the known performance of the cable or device.
2. To verify S21 isolation, connect two loads: one on port 1 and one on port 2. Measure the magnitude of S21 and verify that it is less than the specified isolation (typically less than -80 dB).

Note: To get a more accurate range of expected values for these measurements, consult the analyzer's specifications.

ECal Confidence Check

ECal Confidence Check is a method to check the accuracy of a calibration performed with mechanical standards or an ECal module. The confidence check allows you to measure an impedance state in the ECal module (called the confidence state), and compare it with factory measured data stored in the module.

In order for this test to be valid, the test ports of the ECal module must connect directly to the calibration reference plane (without adapters).

Note: In the **N469x** series of 2-port ECal modules, from the module minimum frequency up to approximately 2 GHz, the confidence state has a very high amount of transmission loss. In this frequency range, calibrated measurements of transmission S-parameters for the confidence state may vary much more than expected from the Agilent-characterized data in the measurement memory trace. When comparing the measurement trace and memory trace you, ignore the data for frequencies up to 2 GHz.

How to Perform ECal Confidence Check:

1. Connect ECal module to the analyzer with the USB cable. See [Connect ECal Module to the PNA](#). **Note:** Terminate any unused ECAL ports with a 50 ohm load.
2. Allow the module to warm up for 15 minutes or until the module indicates **READY**.
3. Do one of the following to start ECal Confidence Check

Using front-panel HARDKEY [softkey] buttons

1. Press **CAL**
2. then **[More]**
3. then **[ECal]**
4. then **[Confidence Check]**

Using a mouse with PNA Menus

1. Click **Response**
2. then **Cal**
3. then **More**
4. then **ECal**
5. then **Confidence Check**

Programming Commands

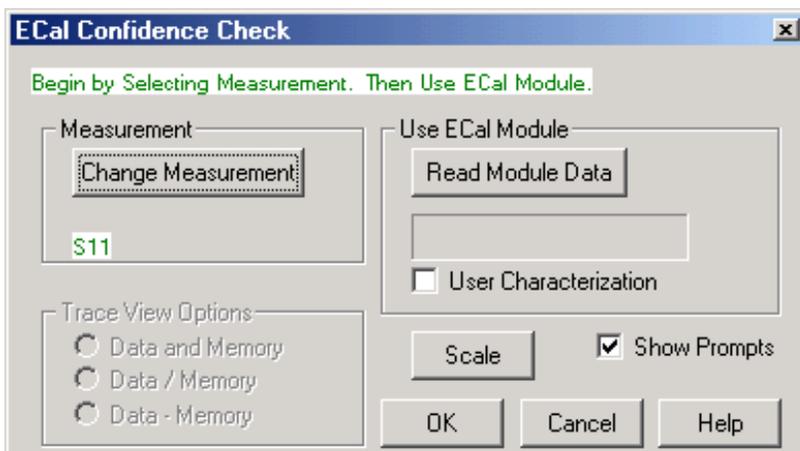
On the following [ECal Confidence Check dialog box](#):

4. Click **Read Module Data**. The following occurs:
 - ECal module is set to "confidence state".
 - PNA reads and displays stored data.
 - PNA measures and displays "confidence state".

5. To view a different parameter, select **Change Measurement** and select the check box for the desired parameter. The default is the active channel parameter.
6. Select the viewing option in the Trace View Options block.
7. Compare the stored and measured data for each measurement parameter.

Notes:

- After exiting ECal Confidence Check, the ECal module remains in the same impedance state and the factory (or user-characterized) data is still stored in the memory trace. Therefore, you can save both the data and memory trace as a *.csv files and import them to a spreadsheet. [Learn how.](#)
- If the two traces show excessive difference, there may be a loose or dirty connection at the test ports or damage to the test cables. Carefully inspect the cables and connections. Then clean and gage each connector, and re-calibrate if needed.
- The User Characterization setting selects the user-characterization data instead of the factory characterization data (available when a User-Characterization is stored in the ECal module).



ECal Confidence Check dialog box help

Compares the accuracy of corrected (calibrated) data with stored data in the ECal module. For the check to be valid, the module test ports must connect directly to the calibration reference plane (without an adapter). [Learn more about ECal Confidence Check.](#)

Measurement

Change Measurement Opens the Measure dialog box.

Use ECal Module

Read Module Data

- Copies stored data from the ECal module to Memory.
- Changes state of ECal module to confidence state.
- Measures and displays confidence state and Memory trace.

User Characterization Selects the user-characterization data (stored in the module) instead of the factory characterization data (available when a User-Characterization is stored in the ECal module).

Scale Opens the Scale dialog box.

Show Prompts Check to show a reminder for the connection (default).

Trace View Options

Data and Memory Trace Displays current measurement data and Memory trace.

Data / Memory Performs an operation where the current measurement data is divided by the data in memory.

Data + Memory Performs an operation where the current measurement data is added to the data in memory.

Verification Kit

Measuring known devices, other than calibration standards, is a straightforward way of verifying that the network analyzer system is operating properly. Verification kits use accurately known verification standards with well-defined magnitude and phase response. These kits include precision airlines, mismatch airlines, and precision fixed attenuators. Traceable measurement data is shipped with each kit on disk and verification kits may be re-certified by Agilent.

See [Analyzer Accessories](#) for a list of Agilent verification kits.

Last modified:

5-May-2010	Added Note
26-May-2009	Added ECal conf check save
3-Sep-2008	Removed legacy content
9/12/06	Added link to programming commands

Calibration Standards

This following section explains the general principles and terms regarding calibration kit files. To learn **how** to modify calibration kit files, See [Modify Calibration Kits](#).

- [About Calibration Kits](#)
- [Calibration Standards](#)
- [Standard Type](#)
- [Standard Definitions](#)
- [Class Assignments](#)

See other Calibration Topics

About Calibration Kits

A calibration kit is a set of physical devices called standards. Each standard has a precisely known or predictable magnitude and phase response as a function of frequency.

In order to calibrate the analyzer using the standards in a calibration kit, the response of each standard must be mathematically defined and then organized into standard classes that correspond to the error models used by the analyzer.

To be able to use a particular calibration kit, the known characteristics from each standard in the kit must be stored into analyzer memory. This is done for you with the PNA. All Agilent Cal Kits containing standard definitions are stored in the PNA. For a list of Agilent calibration kits, see [Analyzer Accessories](#).

Calibration Standards

Calibration standards provide the reference for error-corrected measurements in the network analyzer. Each standard has a precisely known definition that includes electrical delay, impedance, and loss. The analyzer stores these definitions and uses them to calculate error correction terms.

During measurement calibration, the analyzer measures standards and mathematically compares the results with "ideal models" of those standards. The differences are separated into error terms that are later removed from device measurements during error correction. [See Systematic Errors](#).

Standard Type

A standard type is one of four basic types that define the form or structure of the model to be used with that standard. The following are the four basic standard types:

Standard	Terminal Impedance
SHORT	zero ohms
OPEN	infinite ohms
LOAD	system impedance, Z_0
THRU/LINE	no terminal impedance

Learn about other Calibration Standards:

- [Data-Based Standard](#)
- [Sliding Load](#)
- [Offset Load](#)
- [Arbitrary Impedance Load](#)

Standard Definitions

Standard definitions describe the electrical characteristics of the standards and the frequencies they will be used. Standard definitions can be viewed from the [Advanced Modify Cal Kit](#) menu selection.

Standard definitions include:

- **Minimum Frequency** Specifies the minimum frequency the standard is used for calibration.
- **Maximum Frequency** Specifies the maximum frequency the standard is used for calibration.
- **Z_0** Specifies the characteristic impedance of the standard (not the system characteristic impedance or the terminal impedance of the standard).
- **Delay** Specifies a uniform length of transmission line between the standard being defined and the actual calibration plane.
- **Type** Specifies type of standard (SHORT, OPEN, THRU/LINE, LOAD, ARBITRARY).
- **Loss** Specifies energy loss, due to skin effect, along a one-way length of coaxial cable.

Loss model equation:

- The value of loss is entered as ohms/second at 1 GHz.
- To compute the loss of the standard, measure the delay in seconds and the loss in dB at 1 GHz. Then use the following formula:

$$\text{Loss} \left(\frac{\Omega}{S} \right) = \frac{\text{loss (dB)} \times Z_0(\Omega)}{4.3429(\text{dB}) \times \text{delay(s)}}$$

Capacitance model equation:

C0, C1, C2, C3. Specifies the fringing capacitance for the open standard.

- $C = (C0) + (C1 \times F) + (C2 \times F^2) + (C3 \times F^3)$
- (F is the measurement frequency).
- The terms in the equation are defined when specifying the open as follows:
 - C0 term is the constant term of the third-order polynomial and is expressed in Farads.
 - C1 term is expressed in F/Hz (Farads/Hz).
 - C2 term is expressed in F/Hz².
 - C3 term is expressed in F/Hz³.

Inductance model equation:

L0, L1, L2, L3. Specifies the residual inductance for the short standard.

- $L = (L0) + (L1 \times F) + (L2 \times F^2) + (L3 \times F^3)$
- (F is the measurement frequency).
- The terms in the equation are defined when specifying the short as follows:
 - L0 term is the constant term of the third-order polynomial and is expressed in Henries.
 - L1 term is expressed in H/Hz (Henries/Hz)
 - L2 term is expressed in H/Hz².
 - L3 term is expressed in H/Hz³.

Class Assignments

Once a standard is characterized, it must be assigned to a standard "class". A standard class is a group of standards that are organized according to the calibration of the PNA error model.

The number of classes needed for a particular calibration type is equal to the number of error terms being corrected.

A class often consists of a single standard, but may be composed of multiple standards. These may be

required for accuracy or to cover a wide frequency range.

Example: A response calibration requires only one class, and the standards for that class may include an OPEN, or SHORT, or THRU. A 1-port calibration requires three classes. A 2-port calibration requires 10 classes, not including two for isolation.

The number of standards assigned to a given class may vary from one to seven for unguided calibrations. Guided calibrations allow as many standards as needed.

Calibration Classes are assigned in the Advanced Modify Cal Kit menu, [SOLT](#) or [TRL](#) tab.

The different classes used in the PNA:

S11A, S11B, S11C (S22A, S22B, S22C and so forth)

These are the three classes for port 1-reflection calibrations (three classes also for S22 and S33). They are used in the one-port calibrations and the full two-port calibration. They are required in removing the directivity, source match, and reflection tracking errors. Typically, these classes might consist of an open, a short and a load standard for each port.

Transmission and Match (forward and reverse)

These classes are used to perform a full two-port calibration. The transmission class relates primarily to the transmission tracking, while the match class refers to load match. For both of these classes, the typical standard is a thru or delay.

Isolation

The isolation classes are used to perform a full two-port and the TRL two-port calibrations. The isolation classes apply to the forward and reverse crosstalk terms in the PNA error model.

TRL THRU

These are used to perform a TRL two-port calibration. The TRL thru class should contain a thru standard or a short line. If it contains a non-zero length thru standard, then the calibration type is called LRL or LRM.

TRL REFLECT

This class is used to perform a TRL two-port calibration. The TRL reflect class should contain a standard with a high reflection coefficient, typically an open or short. The actual reflection coefficient need not be known, but its phase angle should be specified approximately correctly (± 90 deg). The exact same reflection standard must be used on both ports in the TRL calibration process.

TRL LINE or MATCH

These are used to perform a TRL two-port calibration. The TRL line or match class should contain line

standards, load standards, or both. If a line standard is used, its phase shift must differ from that of the TRL THRU standard by 20° to 160° . This limits the useable frequency range to about 8 to 1. Two or more line standards of different lengths may be specified to get broader frequency coverage. It is also common to include a load standard for covering low frequencies, where the line's length would be impractically long. When a load is used, the calibration type is called TRM or LRM.

Note: For more information, read [Specifying Calibration Standards and Kits for Agilent Vector Network Analyzers \(Application Note 1287-11\)](#)

Modify Calibration Kits

Beginning with A.10.00, VNA CalKit Editor, a standalone application, has been integrated into the PNA Firmware.

The following topics discuss Modifying Calibration Kits:

Using VNA CalKit Editor

- [Connectors Tab](#)
- [Standards Tab](#)
- [SOLT Tab](#)
- [TRL Tab](#)

Concepts

- [Why Modify a Cal Kit](#)
- [VNA Cal Kit File Types](#)
- [About PNA Cal Kits and Firmware Upgrades](#)

Procedures

- [How to Create a New Cal Kit from an Existing Cal Kit](#)
- [Creating Custom Calibration Kits using a New Connector Family](#)

Note: Although the User Interface is changed, the existing remote commands can still be used to modify Cal Kits.

How to Modify Cal Kits

The series of dialog boxes that follow allow you to modify the standard definitions or class assignments of calibration kit files.

Using front-panel HARDKEY [softkey] buttons

1. Press **CAL**
2. then **[More]**
3. then **[Cal Kit]**

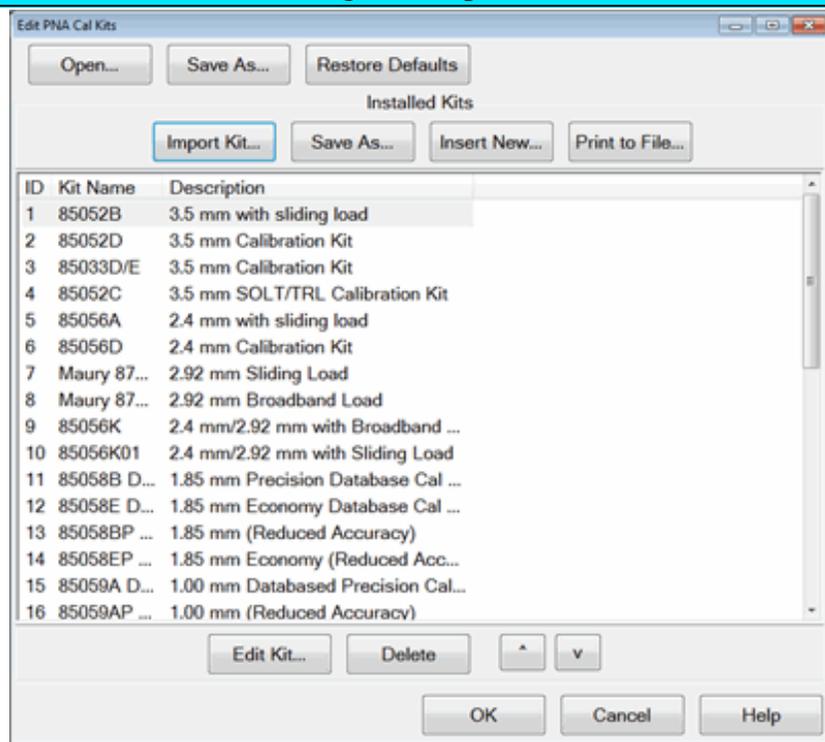
Using a mouse with PNA Menus

1. Click **Response**
2. then **Cal**
3. then **More**
4. then **Cal Kit**

Programming Commands

Edit PNA Cal Kits

Edit PNA Cal Kits dialog box help



Provides access to all Agilent cal kits and allows modification of their standard definitions.

PNA Cal Kits and Firmware Upgrades

- The default "factory" cal kits are overwritten when new firmware is installed. Your custom cal kits (files with custom filenames) are NOT overwritten. However, the custom cal kits must be imported (click **Import Kit**) into the new firmware. [Learn how to import cal kits.](#)
- All PNA cal kits can only be imported by the current firmware revision and later. They can NOT be imported by PAST firmware revisions. Once a Cal Kit has been imported by a later firmware revision, it cannot be imported by the previous version of firmware from which it originated.
- When a firmware upgrade takes place, ALL cal kits, both factory and custom, that are present on the PNA are saved to a single *.wks file using a unique filename. These files are NOT Excel spreadsheet files. They are opened using the **Open** button (see below). They can be used as archives of cal kits from previous firmware versions.

Open Opens an archive of cal kits from past firmware upgrades and 'Save As' operations.

Save As Saves ALL cal kits in the PNA to a *.wks file.

Restore Defaults Re-installs the default factory contents of all Agilent cal kits from the PNA hard drive. The factory Agilent cal kits are stored on the PNA hard drive at c:\users\public\network analyzer\PnaCalKits\factory.

Installed Kits

Import Kit Invokes the [Import Kit](#) dialog box.

Save As Saves the selected calibration kit and definitions (using **.ckt** file type).

Insert New Invokes a blank [Edit Kit dialog box](#) to create new calibration kit definitions.

Print to File Prints the contents of the selected cal kit to a .prn file.

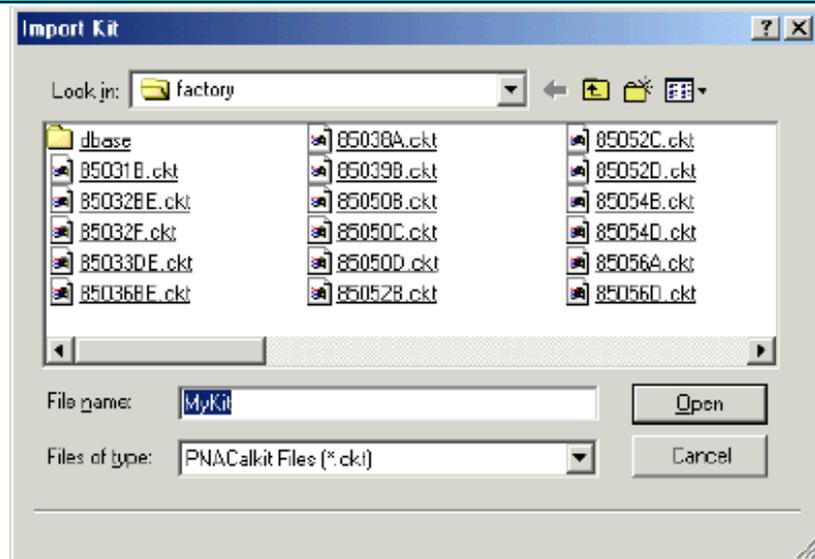
Edit Kit... Invokes the [Connectors tab](#) of the Edit Kit dialog box to modify selected calibration kit definitions.

Delete Deletes selected calibration kit file.

^ Selects previous / next calibration kit in list.

For more information see [Creating Custom Calibration Kits using a New Connector Family.](#)

Import Kit dialog box help



Note: There is no limit to the number of cal kits that can be imported. However, during an [Unguided cal](#), you can access ONLY mechanical cal kits #1 through 95.

Imports calibration kit definitions from hard disk or other drive that are saved in the various formats. The following four cal kit types can be imported.

Files of type Select the file type of your Cal Kit. Learn more about [VNA Cal Kit File Types](#)

File name Navigate and select your cal kit file.

Open Imports the selected file. The kit is added at the end of the list of cal kits.

Note: [See PNA Cal Kits and Firmware Upgrades](#)

Importing Kits other than current PNA Series Kits

Cal kit files from Agilent "legacy" network analyzers (listed above) may not contain information that the PNA requires. Therefore, the PNA may modify the cal kit name and description, the cal standards, and the cal class assignments in a best effort manner. You may need to correct these modifications after importing your legacy cal kit to meet your specific requirements.

- "Legacy" cal kit files are based on the analyzer test port gender while PNA cal kits are based on the Device Under Test (DUT) connector gender. Therefore, when the kit is imported, the standard label and description are reversed and are noted as F- (female) and M- (male) .
- When a Coaxial standard is detected in the kit file, a pair of male/female connectors is typically created.
- Waveguide standards that are created as connector have no gender.

Last modified:

25-Nov-2013	Updated for Win7 locations
25-Jun-2013	Integrated CalKit Editor (10.0)

Why Modify a Cal Kit

For most applications, the default calibration kit models provide sufficient accuracy for your calibration. However, several situations may exist that would require you to create a custom calibration kit:

- Using a connector interface different from those used in the predefined calibration kit models.
 - Using standards (or combinations of standards) that are different from the predefined calibration kits. For example, using three offset SHORTs instead of an OPEN, SHORT, and LOAD to perform a 1-port calibration.
 - Improving the accuracy of the models for predefined kits. When the model describes the actual performance of the standard, the calibration is more accurate. For example: A 7 mm LOAD is determined to be 50.4 instead of 50.0 .
 - Modifying the THRU definition when performing a calibration for a non-insertable device.
 - Performing a TRL calibration.
-

Last modified:

2-Apr-2013 New file

About PNA Cal Kits and Firmware Upgrades

The default "factory" cal kits are overwritten when new firmware is installed. Your custom cal kits (files with custom filenames) are NOT overwritten. However, the custom cal kits must be imported (click **Import Kit**) into the new firmware.

All PNA cal kits can only be imported by the current firmware revision and later. They can NOT be imported by PAST firmware revisions. Once a Cal Kit has been imported by a later firmware revision, it cannot be imported by the previous version of firmware from which it originated.

When a firmware upgrade takes place, ALL cal kits, both factory and custom, that are present on the PNA are saved to a single *.wks file using a unique filename. These files are NOT Excel spreadsheet files. They are opened using the **Open** button (see below). They can be used as archives of cal kits from previous firmware versions.

Last modified:

2-Apr-2013 New file

Creating a New Connector Family

To create a custom calibration kit that uses a new connector type, you must first define the connector family. The connector family is the name of the connector-type of the calibration kit, such as:

- APC7
- 2.4 mm
- Type-N (50)

Although more than one connector family is allowed, it is best to limit each calibration kit to only one connector family.

If you are using a connector family that has male and female connectors, include definitions of both genders. If you are using a family with no gender, such as APC7, only one connector definition is required.

Use the following steps to create a custom calibration kit:

1. In the [Connectors Tab](#), click **Add** to name the new connector family.
2. Enter the Kit Description for the custom cal kit.
3. Click **Add** in the Connectors section of the dialog box.
4. Enter a Connector Family name.
5. Enter a Description of the connector.
6. Select the Gender of one of the connectors.
7. Enter the minimum and maximum Frequency Range.
8. Enter the Impedance.
9. Click the down-arrow to select the Media.
10. Enter the cut-off frequency
11. Click **Apply**.
12. Click **OK**.
13. If you need to add another connector gender, in the [Connectors Tab](#), click **Add** in the Connectors section again for the next connector gender.
14. If you are adding another connector gender, repeat step 3.

Note: If you have male and female versions of the connector family, you probably do NOT also have a NO GENDER version.

Enter Standards

Now that the connector family is added to the custom cal kit, you are ready to add new calibration standards.

1. In the [Standards Tab](#), under the list of standards, click **Add**.
2. Select the type of standard (OPEN, SHORT, LOAD, or THRU), then click **OK**.
3. Complete the information in the dialog box for the standard you selected. Note that for banded standards, the start and stop frequency may be different than the frequency range of the specified connector. Edit the start and stop frequencies as needed. Click **OK** when all the settings are correct.
4. Repeat steps 2 - 3, as necessary, to add all standards and definitions to the new custom cal kit.
5. Assign each of the standards to a calibration class. This is done through the [TRL Tab](#) or [SOLT Tab](#)
6. Save the Cal Kit.

Last modified:

2-Apr-2013 New file

How to Create a New Cal Kit from an Existing Cal Kit

You can create a new custom Cal Kit using an existing Cal Kit as a starting point.

Here is how:

1. From the [Edit Cal Kits](#) dialog, load the Cal Kit you want to use as a starting point.
2. Immediately click **Save As** and change the file name. Select either ***.xkt** or ***.ckt** file type. [Learn more about these file types.](#)
3. Make modifications to your new custom Cal Kit as required.
4. Routinely save your work by clicking **Save**.

See Also

[About PNA Cal Kits and Firmware Upgrades](#)

Last modified:

2-Apr-2013 New file

VNA Cal Kit File Types

The VNA Cal Kit editor can open the following types of Cal Kit files:

VNA Families	File Type
Current PNA Series Cal Kits	*.xkt (default with A.10.00 and above) *.ckt
Old PNA Series Cal Kit (before A.07.50)	*.ck1
8510 Cal Kit	.CK_*
8753, 8752, 8719, 8720, or 8722 Cal Kit	*.ck
Current FieldFox Cal Kits	*.xkt
Previous FieldFox Cal Kits	*.xml

Note: Currently, the VNA Cal Kit editor can NOT open or save to ENA file type ***.ckx**

The current revision of Cal Kit files can be downloaded at <http://na.tm.agilent.com/pna/caldefs/stddefs.html>

File Save (As)

The VNA Cal Kit Editor saves Cal Kits in two formats:

- ***.xkt** - A custom format that is based on xml.
- ***.ckt** - PNA format - can be read by PNA Rev. A.07.50 and higher.

File Association

With the exception of *.xml, the above file types are automatically associated with the CalKit Editor if they are not already associated with a different program. That means, after running CalKit Editor, double-clicking any of the above file types (except *.xml) will open the file using CalKit Editor.

If you have already associated one of these file types with a different program and would like to change it to CalKit Editor, do the following:

1. Right-click the file, then click **Open With**
2. Browse to the CalKitEditor install folder.
 - For a 32-bit version of Windows: C:\Program Files\Agilent\VNA Cal Kit Editor

- For a 64-bit version of Windows: C:\Program Files (x86)\Agilent\VNA Cal Kit Editor

3. Check **Always use the selected program to open this kind of file.**
4. Select **CalKit Editor.**

About Opening Legacy VNA Kits

Cal kit files from Agilent "legacy" network analyzers (listed above) may not contain information that the VNA requires. When loaded into the VNA Cal Kit Editor, the cal kit name and description, the cal standards, and the cal class assignments will be modified in a best effort manner. You may need to correct these modifications after importing your legacy cal kit to meet your specific requirements.

- "Legacy" cal kit files are based on the analyzer test port sex. Modern VNA cal kits are based on the Device Under Test (DUT) connector sex. Therefore, when the kit is imported the standard's label and description are reversed and are noted as F- (female) and M- (male) .
- When a Coaxial standard is detected in the kit file, a pair of male/female connectors is typically created.
- Waveguide standards that are created as connector have no gender.

Last modified:

2-Apr-2013 New file

Connectors Tab

The screenshot shows a software interface for configuring connectors. At the top, there are tabs for 'Connectors', 'Standards', 'SOLT', and 'TRL'. The 'Connectors' tab is active. Below the tabs, there is a 'Connector Family' dropdown menu set to '3.5 mm', with 'Add' and 'Delete' buttons to its right. The 'Frequency Range' section has 'Min' and 'Max' input fields, both set to '0' and '999000' respectively, with 'MHz' units. The 'Gender' section has two radio buttons: 'Gendered' (selected) and 'Genderless'. The 'Impedance' section has a 'Z0' input field set to '50' with 'ohms' units. The 'Transmission Media' section has a 'Media' dropdown menu set to 'WAVEGUIDE', and 'Cutoff Frequency' and 'Height/Width Ratio' input fields, both set to '0' with 'MHz' units.

Cal Kit Name Allows you to change the Name of the selected calibration kit.

Cal Kit Description Allows you to change the description of the selected calibration kit.

Connector Family .Click the down arrow to select the connector family associated with the Cal Kit.

Add Starts the [Add Connector](#) dialog box which allows you to add new connector type to the calibration kit.

Delete Deletes - **WITHOUT WARNING** - the selected connector family.

Note: To modify a connector family or name, Add a new connector, then delete the old connector.

The following is the list of Factory-defined connector type strings:

	Type N (50) female	7-16 female	X-band waveguide
	Type N (50) male	7-16 male	P-band waveguide
APC 3.5 female	Type N (75) female	2.92 mm female	K-band waveguide
APC 3.5 male	Type N (75) male	2.92 mm male	Q-band waveguide
APC 2.4 female	Type F (75) female	1.85 mm female	R-band waveguide
APC 2.4 male	Type F (75) male	1.85 mm male	U-band waveguide
APC 7	Type A (50) female	1.0 mm female	V-band waveguide
	Type A (50) male	1.0 mm male	W-band waveguide
	Type B		

Frequency Range

Min Allows you to define the lowest frequency at which the standard is used for calibration.

Max Allows you to define the highest frequency at which the standard is used for calibration.

Gender

Gendered - The connector family contains both Male and Female connectors.

Genderless - The connector family does NOT contain Male and Female connectors. APC7 connectors are an example of this connector type.

Impedance

Specify the impedance of the standard.

Media

The medium (or 'geometry') of the connector (COAX or WAVEGUIDE).

Cutoff Frequency If Media is Waveguide, type the low-end cutoff frequency.

Height/Width Ratio Used to calculate waveguide loss. This value is usually on the data sheet for waveguide devices.

About Waveguide Cal Kits

If modifying or creating a waveguide cal kit, be sure to make the following settings. You can [create a](#)

[custom waveguide cal kit](#) using an existing factory waveguide Cal kit as a starting point. The factory cal kits already have these settings.

- Frequency Range: **Min. frequency = Cutoff frequency.**
- Gender: **No Gender**
- Impedance Z0: **1 ohm**
- Media: **Waveguide**

For waveguide, choose TRL (Thru-Reflect-Line) calibration type . These calibration types are more accurate and take fewer steps than SOLT.

Add Connector Family



Enter a name for the new connector family. Then click **OK**.

Available at the bottom of every tab

Save As - Allows you to save the cal kit to a new file name and type.

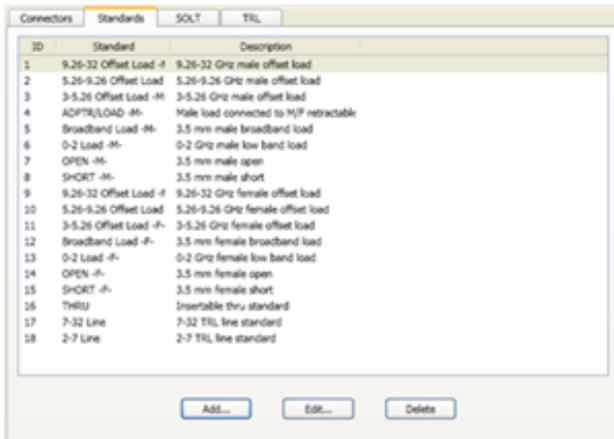
Save - Saves the cal kit to the same file name and type.

Close - Closes the cal kit editing session. The file is NOT saved automatically.

Last modified:

2-Apr-2013 New file

Standards Tab



ID	Standard	Description
1	9.26-32 Offset Load -f	9.26-32 GHz male offset load
2	5.26-9.26 Offset Load	5.26-9.26 GHz male offset load
3	3-5.26 Offset Load -M	3-5.26 GHz male offset load
4	ADPTR/LOAD -M-	Male load connected to M/F retractable
5	Broadband Load -M-	3.5 mm male broadband load
6	0-2 Load -M-	0-2 GHz male low band load
7	OPEN -M-	3.5 mm male open
8	SHORT -M-	3.5 mm male short
9	9.26-32 Offset Load -f	9.26-32 GHz female offset load
10	5.26-9.26 Offset Load	5.26-9.26 GHz female offset load
11	3-5.26 Offset Load -f-	3-5.26 GHz female offset load
12	Broadband Load -f-	3.5 mm female broadband load
13	0-2 Load -f-	0-2 GHz female low band load
14	OPEN -f-	3.5 mm female open
15	SHORT -f-	3.5 mm female short
16	THRU	Insertible thru standard
17	7-32 Line	7-32 TRL line standard
18	2-7 Line	2-7 TRL line standard

Allows you to Add, Edit or Delete cal standards in a cal kit.

Add Standard (Open, Short, Load, Thru, or Data-based)

Add Standard dialog box help



Select the type of standard to be added:

OPEN

SHORT

LOAD

THRU

DATA-BASED

ISOLATION

OK Cancel Help

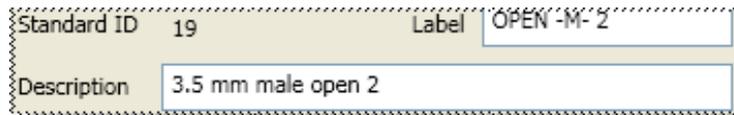
Allows you to add standards to the calibration kit file.

Choose from:

- [OPEN](#)
- [SHORT](#)
- [LOAD](#)
- [THRU](#)
- [DATA-BASED](#)
- [ISOLATION](#)

Standards dialog box help

The following fields apply to **ALL** standard types:



A screenshot of a dialog box with a light beige background and a dashed border. It contains three fields: 'Standard ID' with the value '19', 'Label' with the value 'OPEN -M- 2', and 'Description' with the value '3.5 mm male open 2'.

The other areas of the dialog change depending on the type of standard selected.

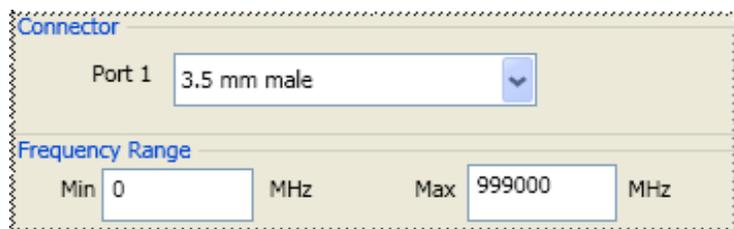
Identification

Standard ID Number in list of standards

Label Type of standard. This usually appears in prompts for standards.

Description Description of standard.

The following fields apply to ALL standard types EXCEPT the Isolation type:



A screenshot of a dialog box with a light beige background and a dashed border. It contains two sections: 'Connector' with a dropdown menu showing '3.5 mm male' and 'Port 1' next to it, and 'Frequency Range' with 'Min' set to '0' and 'Max' set to '999000', both followed by 'MHz' labels.

Connector

Indicates the type and gender (Male, Female, None) of the standard.

Thru and Isolation standards have two connectors.

Data-Based standards MAY have two connectors.

Frequency Range

Min Defines the lowest frequency at which the standard is used for calibration.

Max Defines the highest frequency at which the standard is used for calibration.

The Delay Characteristics fields apply to MOST standard types:

Delay Characteristics

Delay pSec Loss Gohms/s

Z0 ohms

Delay Characteristics

Delay Defines the one-way travel time from the calibration plane to the standard in seconds.

Z0 Defines the impedance of the standard.

Loss Defines energy loss in Gohms, due to skin effect, along a one-way length of coaxial cable.

Other fields are unique to standard type

Choose from:

- [OPEN](#)
- [SHORT](#)
- [LOAD](#)
- [THRU](#)
- [DATA-BASED](#)
- [ISOLATION](#)

Open Standard

C0, C1, C2, C3 Specifies the fringing capacitance.

Open Characteristics

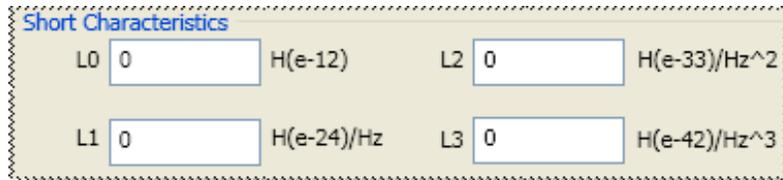
C0 F(e-15) C2 F(e-36)/Hz²

C1 F(e-27)/Hz C3 F(e-45)/Hz³

These are the unique fields of the dialog. [See the areas that are common to all standards.](#)

Short Standard

L0, L1, L2, L3 Specifies the residual inductance.

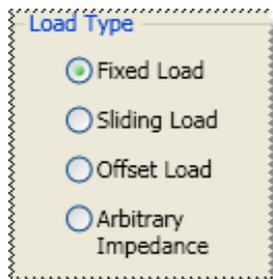


Short Characteristics

L0	<input type="text" value="0"/>	H(e-12)	L2	<input type="text" value="0"/>	H(e-33)/Hz^2
L1	<input type="text" value="0"/>	H(e-24)/Hz	L3	<input type="text" value="0"/>	H(e-42)/Hz^3

Load Standard

Choose from the following



Load Type

- Fixed Load
- Sliding Load
- Offset Load
- Arbitrary Impedance

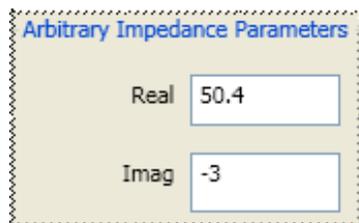
Fixed Load Specifies the load type as Fixed. The fixed load is assumed to be a perfect termination without reflection.

Sliding Load

A sliding load is defined by making multiple measurements of the device with the sliding load element positioned at various marked positions of a long transmission line. The transmission line is assumed to have zero reflections and the load element has a finite reflection that can be mathematically removed using a least squares circle fitting method.

A sliding load cal can be very accurate when performed perfectly. It can also be very inaccurate when not using proper technique. **For accurate results, closely follow the users manual instructions for the sliding load.**

Arbitrary Impedance



Arbitrary Impedance Parameters

Real	<input type="text" value="50.4"/>
Imag	<input type="text" value="-3"/>

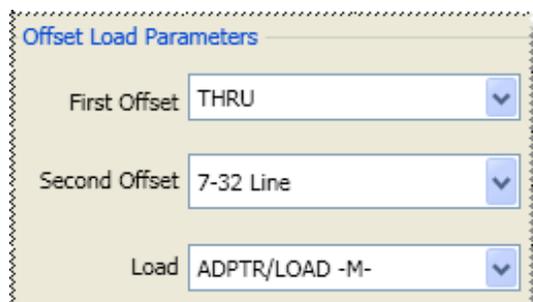
Specifies the load type that has an impedance value different from system Z0. An arbitrary impedance device is similar to a fixed load except that the load impedance is NOT perfect. Early

firmware releases of the PNA series used a fixed resistance value. A complex terminating impedance has been added to allow for more accurate modeling of circuit board or on-wafer devices.

The following Complex Impedance settings are available ONLY when Arbitrary Impedance is selected.

- **Real** The real portion of the impedance value.
- **Imaginary** The imaginary portion of the impedance value.

Offset Load



Offset Load Parameters

First Offset THRU

Second Offset 7-32 Line

Load ADPTR/LOAD -M-

In Jan 2006, Offset Load definitions were added to TRL and Waveguide Cal Kit files. Using an Offset Load standard results in a more accurate calibration than with a Broadband Load. Therefore, when performing a calibration using one of the modified Cal Kit definitions, you may be prompted to connect more standards than before this change. To revert to using the Broadband Load Standard without offset, do the following:

1. Click **Calibration**, then **Advanced Modify Cal Kit**
2. Select the kit, then click **Edit Kit**
3. Under Class Assignments, click **Edit**
4. Select Calibration Kit Class **S11C** (Loads)
5. Under Selected Standards, select **Broadband Load**, then click **Move Up** until the standard is at the top of the list. This will ensure that the Broadband Load is used first.

About Offset Load

An offset load is a compound standard consisting of a load element and two known offset elements (transmission lines) of different length. The shorter offset element can be a zero-length (Flush-thru) offset. The load element is defined as a 1-port reflection standard. An offset load standard is used when the response of the offset elements are more precisely known than the response of the load element. This is the case with waveguide. Measurement of an offset load standard consists of two

measurements, one with each of the two offset elements terminated by the load element. The frequency range of the offset load standard should be set so that there will be at least a 20 degree separation between the expected response of each measurement.

To specify more than two offset elements, define multiple offset load standards. In cases where more than two offsets are used, the frequency range may be extended as the internal algorithm at each frequency will search through all of the possible combinations of offsets to find the pair with the widest expected separation to use in determining the actual response of the load element.

The following Offset Load settings are available ONLY when Offset Load is selected.

- First Offset Standard
- Second Offset Standard
- Load Standard

Thru Standard

The screenshot shows a configuration dialog for a Thru Standard. It is divided into several sections:

- Connectors:** Port 1 is set to "3.5 mm female" and Port 2 is set to "3.5 mm male".
- Frequency Range:** Min is 0 MHz and Max is 38800 MHz.
- Delay Characteristics:** Delay is 0 pSec, Loss is 0 Gohms/s, and Z0 is 50 ohms.
- Virtual Device:** A checkbox labeled "Virtual Device" is checked.

Connectors - Defines connector type at both ends of the Thru standard.

Virtual Device

Most cal kits have only one Thru standard definition for SOLT calibrations. For these cases, use the default selections (checked for zero-length Thrus and cleared for non-zero-length Thrus).

This checkbox is used to make forward and reverse measurements of your Thru standard for the same pair of ports in two separate steps. This is NOT common for zero-length (Flush) Thru standards.

When **checked**, calibration prompts involving that Thru will **omit** the Description. For example “Connect port 1 to port 2”. This is the common prompt for Flush-Thru standards.

When **cleared**, calibration prompts for that Thru will **include** the Description. For example “Connect <standard description> between ports 1 and 2”.

To make forward and reverse measurements of your Thru standard for the same pair of ports as two separate steps, do the following:

1. Create separate definitions of the Thru standard(s) using the same settings, except for the Label and Description. **Clear** this checkbox for BOTH definitions.
2. For one Thru definition, in the label and description include the word 'FORWARD' to prompt the operator to use this standard for the forward measurement. Assign this standard to the SOLT “FWD TRANS” and “FWD MATCH” classes of the cal kit.
3. For the Thru definition, in the label and description include the word 'REVERSE' to prompt the operator to use this standard for the reverse measurement. Assign this standard to the SOLT “REV TRANS” and “REV MATCH” classes of the cal kit.
4. When you perform SOLT calibrations using this cal kit, the forward measurements of the Thru will be measured in one connection step, and the reverse measurements in another.

Data-Based Standard

Response Data Summary

Number of Response Data Variables = 4
index = 0: Number of Data Variable Values = 900
Data Variable Name = S11
index = 1: Number of Data Variable Values = 900
Data Variable Name = S21

Load Data File...

Accuracy Data Summary

Standard contains no accuracy data

Load Data File...

Virtual Device

Note: To learn how to modify data-based standard files, visit <http://na.tm.agilent.com/pna/dbcal.html>

Learn about the relative accuracy of [Databased versus Polynomial Cal Kits](#).

The modified file can then be uploaded into the VNA.

Upload Data From File

Click **Load Data File**, then navigate to the *.dat file which is provided with the data-based Cal Kit. Both Response data and Accuracy (Uncertainty) data is provided in a single *.dat file.

For Advanced Users

Response data can be loaded from a *.s2p or *.cti file.

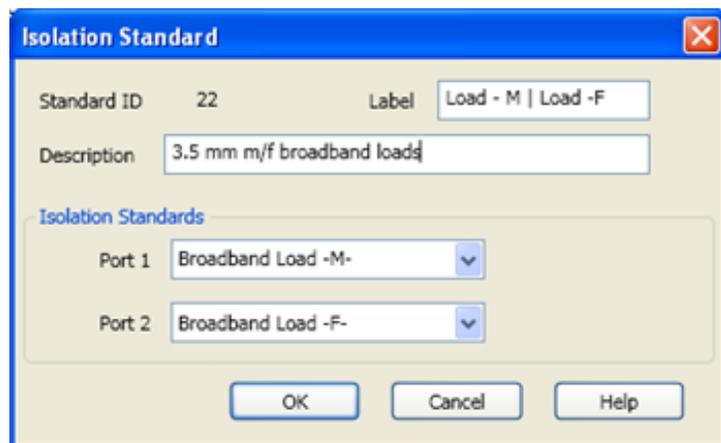
Accuracy data can be loaded from a *.cti file.

Virtual Device

This checkbox is displayed for a Data-Based cal standard when the standard has been defined to have 2 ports.

- When Cleared (default) calibration prompts for that standard will include it's Description. For example "Connect <standard description> between ports 1 and 2".
- When Checked, calibration prompts for that standard will NOT include its Description, so the prompt will be worded as if the data-based standard is a zero-length Thru connection. For example "Connect port 1 to port 2".

Isolation Standard



The screenshot shows a dialog box titled "Isolation Standard". It has a blue title bar with a close button. The main area is light beige. At the top, there are two fields: "Standard ID" with the value "22" and "Label" with the value "Load - M | Load -F". Below these is a "Description" field containing "3.5 mm m/f broadband loads". Underneath is a section titled "Isolation Standards" which contains two dropdown menus: "Port 1" with the selected value "Broadband Load -M-" and "Port 2" with the selected value "Broadband Load -F-". At the bottom of the dialog are three buttons: "OK", "Cancel", and "Help".

The pair of loads are considered one standard.

Both loads are connected to the VNA and measured with the same prompt.

Available at the bottom of every tab

Save As - Allows you to save the cal kit to a new file name and type.

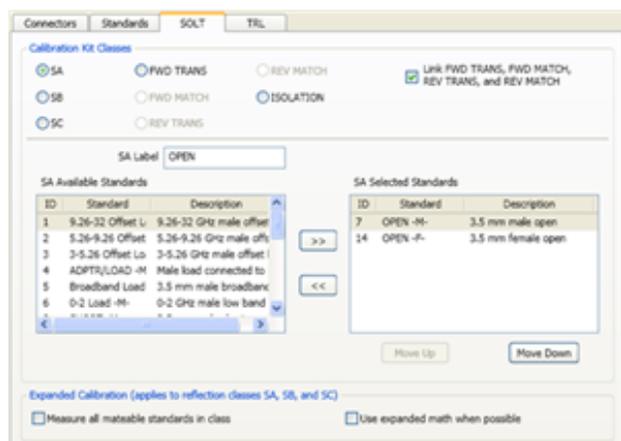
Save - Saves the cal kit to the same file name and type.

Close - Closes the cal kit editing session. The file is NOT saved automatically.

Last modified:

2-Apr-2013 New file

SOLT Tab



Note: This dialog looks similar to the dialog that appears after the [Cal Wizard View / Modify dialog](#). However, setting changes in that dialog affect **ONLY** the calibration that is in progress. These settings, accessed through [Modify Cal Kit](#), changes the cal kit for all future calibrations that use this cal kit.

Allows you to assign single or multiple standards to **SOLT** Calibration Classes.

Click the [TRL tab](#) to assign standards to TRL Calibration Classes.

1. For each Cal Kit Class, select **Available Standards** from the left list, then click **>>** to copy the standard to the Cal Kit.
2. Use **MOVE UP** and **MOVE DOWN** to change the **ORDER** of the standards. The order is used in guided calibrations to determine which standards in that class will be used in calibrations that involve the frequency ranges over which the standards are defined. Guided cals will include standards in the order in which they appear in this class list, and in the case where standards in the class list have frequency ranges that overlap, the order also determines which standard is used for frequencies in the overlap range. For example, let's assume that you define a broadband Short from Min Freq.= 0 Hz and Max Freq.= 999 GHz, and that standard is listed first in the SB or TRL REFLECT class. If you then list a frequency-banded Short with the same connector below the broadband short in those same classes, then guided calibrations would not use the frequency-banded Shorts because the broadband Shorts would always be given priority.

SOLT <cal class> Label

The cal standard category label that appears in the VNA's user interface during **unguided** SOLT calibrations.

Calibration Kit Classes

For each calibration class, select **Available Standards**, then click **>>** to move to the **Selected Standards** list.

- **SA** - OPEN Standards
- **SB** - SHORT Standards
- **SC** - LOAD Standards
- **FWD / REV Trans and Match** - THRU Standards. Most Cal Kits do NOT include a physical THRU standard, but assume that an Insertable Thru will be used.
- ISOLATION - Isolation standard. For PNA analyzers, ISOLATION calibration is not usually recommended. It could be beneficial in some situations where custom user-supplied test set hardware is being used.

Note: In older VNA models, the SA, SB, and SC classes were labeled S11A, S11B, S11C and S22A, S22B, S22C. Separate classes for ports 1 and 2 is now considered unnecessary, especially in the context of VNA models with more than 2 ports. When saving a cal kit in the PNA *.ckt format with this editor and viewing the kit in PNA firmware with the older user interface, the standard selections for those three classes are duplicated in both the S11 and S22 selections for those classes.

Link FWD TRANS, FWD MATCH, REV TRANS, and REV MATCH Check to automatically assign the standard definition for FWD TRANS to FWD MATCH, REV MATCH, and REV TRANS. Clear to separately assign FWD MATCH, REV MATCH and REV TRANS classes (SOLT calibrations only).

Expanded Calibration

The following two check boxes **apply ONLY during Guided Calibrations**. For Unguided Calibration, these check boxes are ignored, including the case where the multiple standards dialog box is presented.

Measure all mateable standards in class Check this box to attain the very highest accuracy possible. For example, if a cal kit contains several load standards, during the calibration process you will be prompted to measure each of the standards. This could require a significant amount of calibration time. When checked, the "Use expanded math when possible" box is also checked automatically.

Use expanded math when possible Some kits contain multiple calibration standards of the same type that together cover a very wide frequency range. (For example: multiple shorts, or a lowband load and a sliding load.) If a calibration requires more than one standard to cover the calibration frequency range, there can be regions of overlapping measurements. When this checkbox is selected, the VNA automatically computes the most accurate measurement in the overlap regions using a "weighted least squares fit" algorithm. This function improves accuracy without slowing the calibration speed.

- Manually select this checkbox only when using a cal kit that contains multiple standards of the

same type. (For example: multiple shorts, or a lowband load and a sliding load.)

- The checkbox is cleared by default when a polynomial model is selected from the cal kit menu.
- The checkbox is selected by default when the 85058B or 85058E data-based model is selected from the cal kit menu.

Available at the bottom of every tab

Save As - Allows you to save the cal kit to a new file name and type.

Save - Saves the cal kit to the same file name and type.

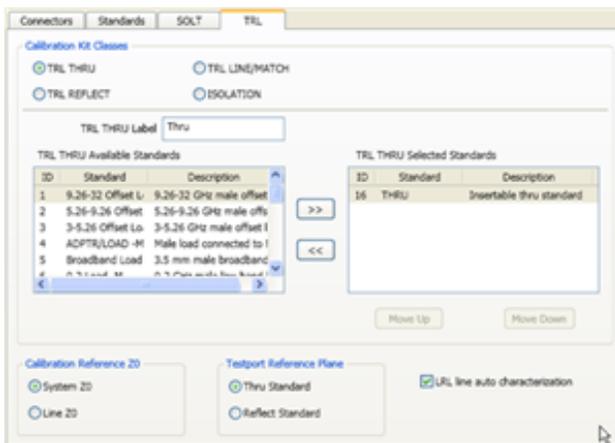
Close - Closes the cal kit editing session. The file is NOT saved automatically.

Last modified:

2-Apr-2013 New file

TRL Tab

On the image below, click a setting area to learn more.



Note: This dialog will look different if you clicked Help at the Cal Wizard View / Modify dialog. The functionality is the same.

Allows you to assign single or multiple standards to Calibration Classes.

1. For each Cal Kit Class, select **Available Standards** from the left list, then click >> to copy the standard to the Cal Kit.
2. Use **MOVE UP** and **MOVE DOWN** to change the **ORDER** of the standards. The order is used in guided calibrations to determine which standards in that class will be used in calibrations that involve the frequency ranges over which the standards are defined. Guided cals will include standards in the order in which they appear in this class list, and in the case where standards in the class list have frequency ranges that overlap, the order also determines which standard is used for frequencies in the overlap range. For example, let's assume that you define a broadband Short from Min Freq.= 0 Hz and Max Freq.= 999 GHz, and that standard is listed first in the SB or TRL REFLECT class. If you then list a frequency-banded Short with the same connector below the broadband short in those same classes, then guided calibrations would not use the frequency-banded Shorts because the broadband Shorts would always be given priority.

Note: The TRL LINE/MATCH class has a slight exception to these prioritization behaviors. In general, Line standards are given a higher priority than Match standards. So if a Line standard and a Match standard are defined to have the same frequency range and the Match standard is listed above the Line standard in the class order, a guided TRL cal will still prefer to use the Line standard rather than the Match standard.

TRL <cal class> Label

The cal standard category label that appears in the VNA's user interface during **unguided** TRL calibrations.

Cal Kit Classes

- For PNA analyzers, ISOLATION calibration is not usually recommended. It could be beneficial in some situations where custom user-supplied test set hardware is being used.

TRL THRU

Note: All **THRU calibration methods** are supported in a TRL Cal **EXCEPT** Unknown Thru.

- The THRU standard can be either a zero-length or non-zero length. However, a zero-length THRU is more accurate because it has zero loss and no reflections, by definition.
- The THRU standard cannot be the same electrical length as the LINE standard.
- If the insertion phase and electrical length are well-defined, the THRU standard may be used to set the reference plane.
- The THRU standard and LINE standard have the same characteristic impedance and are perfectly matched. They define the reference impedance of the calibration.
- If a THRU standard with the correct connectors is NOT available, an adapter removal cal can be performed.

TRL REFLECT

- The REFLECT standard can be anything with a high reflection, as long as it is the same when connected to one or more VNA ports.
- The REFLECT standard on each port is identical.
- The actual magnitude of the reflection need not be known.
- The phase of the reflection standard must be known within 1/4 wavelength.
- If the magnitude and phase of the reflection standard are well-defined, the standard may be used to set the reference plane.

TRL LINE

The LINE and THRU standards establish the reference impedance for the measurement after the calibration is completed. TRL calibration is limited by the following restrictions of the LINE standard:

- Must be of the same impedance and propagation constant as the THRU standard.
- The electrical length need only be specified within 1/4 wavelength.
- Cannot be the same length as the THRU standard.
- A TRL cal with broad frequency coverage requires multiple LINE standards. For example, a span from 2 GHz to 26 GHz requires two line standards.

- Must be an appropriate electrical length for the frequency range: at each frequency, the phase difference between the THRU and the LINE should be greater than 20 degrees and less than 160 degrees. This means in practice that a single LINE standard is only usable over an 8:1 frequency range (Frequency Span / Start Frequency). Therefore, for broad frequency coverage, multiple lines are required.
- At low frequencies, the LINE standard can become too long for practical use. The optimal length of the LINE standard is 1/4 wavelength at the geometric mean of the frequency span (square root of $f_1 \times f_2$).

Note: The TRL LINE standard must have a delay that is greater than 0 (zero) ps. Otherwise, calibration correction calculations will contain unpredictable results.

TRL MATCH

If the LINE standard of appropriate length or loss cannot be fabricated, a MATCH standard may be used instead of the LINE.

- The MATCH standard is a low-reflection termination connected to both Port 1 and Port 2.
- The MATCH standard may be defined as an infinite length transmission line OR as a 1-port low reflect termination, such as a load.
- When defined as an infinite length transmission line, both test ports must be terminated by a MATCH standard at the same time. When defined as a 1-port load standard, the loads are measured separately. The loads are assumed to have the same characteristics.
- The impedance of the MATCH standard becomes the reference impedance for the measurement. For best results, use the same load on both ports. The load may be defined using the data-based definition, the arbitrary impedance definition, or the fixed load definition.

Calibration Reference Z0 (TRL only)

System Z0 The system impedance is used as the reference impedance. Choose when the desired test port impedance differs from the impedance of the LINE standard. Also, choose when skin effect impedance correction is desired for coax lines.

Line Z0 The impedance of the line standard is used as the reference impedance, or center of the Smith Chart. Any reflection from the line standard is assumed to be part of the directivity error.

Testport Reference Plane (TRL only)

Thru Standard The THRU standard definition is used to establish the measurement reference plane. Select if the THRU standard is zero-length or very short.

Reflect Standard The REFLECT standard definition is used to establish the position of the measurement reference plane. Select if the THRU standard is not appropriate AND the delay of the REFLECT standard is well defined.

Also, select If a flush short is used for the REFLECT standard because a flush short provides a more accurate phase reference than a Thru standard.

LRL line auto characterization

Note: This setting ONLY applies if an LRL Cal Kit is being modified **AND** Testport Reference Plane is set to Thru Standard **AND** the TRL Thru class standard and the TRL Line/Match class standard both have the same values for Offset Z0 and Loss. Otherwise, this setting is ignored.

- Check the box to allow the VNA to automatically correct for line loss and dispersion characteristics.
- Clear the box if anomalies appear during a calibrated measurement which may indicate different loss and impedance values for the Line standards.

Available at the bottom of every tab

Save As - Allows you to save the cal kit to a new file name and type.

Save - Saves the cal kit to the same file name and type.

Close - Closes the cal kit editing session. The file is NOT saved automatically.

Last modified:

2-Apr-2013 New file

Markers

Markers provide a numerical readout of measured data, a search capability for specific values, and can change stimulus settings. There are 9 regular markers and one [Reference marker](#) (used with Delta markers) available per trace. This topic discusses all aspects of markers.

Note: Marker Readout can be turned ON / OFF and customized from the **View/Display** menu. [Learn more.](#)

- [Creating and Moving Markers](#)
- [Marker Dialog](#)
- [Searching with Markers](#)
- [Marker Functions](#) (Change Instrument Settings)
- [Marker Display](#)
- [Marker Table](#)

Other Analyze Data topics

How to Create Markers

Using front-panel HARDKEY [softkey] buttons

1. Press **MARKER**
2. then **[Marker n]**

Using a mouse with PNA Menus

1. Click **Marker/Analysis**
2. then **Marker**
3. select a marker number

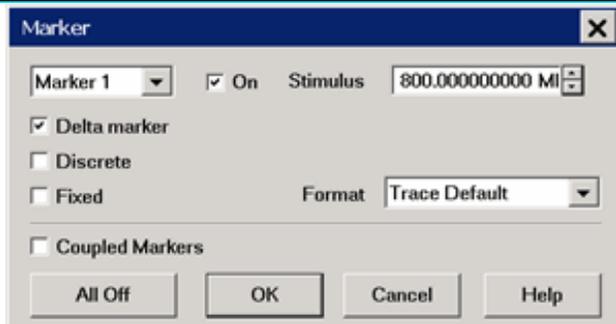
[Programming Commands](#)

Moving a Marker

To move a marker, make the marker active by selecting its number in any of the previous 3 methods. The **active marker** appears on the analyzer display as ∇ . All of the other markers are inactive and are represented on the analyzer display as Δ . Then change the stimulus value using any of the following methods:

- Type a value.
- Scroll to a stimulus value using the up / down arrows. The resolution can not be changed.
- Click the stimulus box, then use the front-panel knob.
- Click and Drag Markers using a finger (touchscreen) or by left-clicking and holding a marker symbol. Then drag the marker to any point on the trace. This feature is NOT allowed in Smith Chart or Polar [display formats](#) or with a [Fixed Marker type](#).

Marker dialog box help



Marker Specifies the current (active) marker number that you are defining.

On Check to display the marker and corresponding data on the screen.

Stimulus Specifies the X-axis value of the active marker. To change stimulus value, type a value, use the up and down arrows, click in the text box and use the front-panel knob, or drag the marker on the screen.

Delta Marker Check to make the active marker display data that is relative to the reference (R) marker. There is only one reference marker per trace. All nine other markers can be regular markers or delta markers. When a delta marker is created, if not already displayed, the reference marker is displayed automatically. A delta marker can be activated from the [Marker dialog box](#) or the [Marker Toolbar](#).

Discrete Marker Check to display values at only the discrete points where data is measured. Clear to display values that are interpolated from the data points. The interpolated marker will report y-axis data from ANY frequency value between the start and stop frequency.

Fixed Check to cause the marker to have a fixed X-axis and **Y-axis** position based on its placement on the trace when it was set to fixed. It does NOT move with trace data amplitude. It can be scrolled left and right on the X-axis by changing the marker stimulus value. Use this marker type to quickly monitor "before and after" changes to your test device. For example, you could use fixed markers to record the difference of test results before and after tuning a filter.

Clear the box to create a **Normal** marker, which has a fixed stimulus position (X-axis) and responds to changes in data amplitude (Y-axis). It can be scrolled left and right on the X-axis by changing the marker stimulus value. Use this marker type with one of the marker search types to locate the desired data.

Coupled Markers Check to couple markers by marker number, 1 to 1, 2 to 2 and so forth. The markers will remain coupled until this box is unchecked. [Learn more about coupled markers.](#)

(Marker) Format Displays the marker data in a format that you choose. The Trace Default setting has the same marker and grid formats. Choose from the following:

Log/Phase	Log Mag	Real
Linear/Phase	Linear Mag	Imaginary
Real/Imag	Phase	Kelvin
R+jX (complex impedance)	SWR	Fahrenheit
G+jB (complex admittance)	Delay	Celsius

All Off Switches OFF all markers on the active trace.

Searching with Markers

You can use markers to search and return data for the following trace criteria:

- [Max and Min](#): find the highest or lowest points on the trace
- [Peak](#), then move to other peaks (left, right, next highest)
- [Target Value](#): find a specific Y-axis value
- [Bandwidth](#) (Filters)
- [Compression Point](#) (Amplifiers)
- [About PSAT and PNOP Markers](#)
 - [Power Saturation](#) (Amplifiers)
 - [Power Normal Operating Point](#) (Amplifiers)
- [What is a Peak?](#)
- [Search Domain](#)
- [Search Within](#)

How to Search with Markers

Using front-panel HARDKEY [softkey] buttons

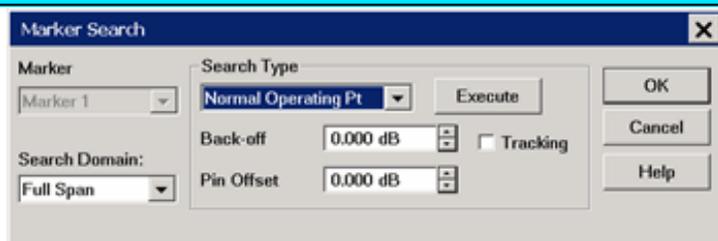
1. Press **SEARCH**

Using a mouse with PNA Menus

1. Click **Marker/Analysis**
2. then **Marker Search**

← Programming Commands →

Marker Search dialog box help



Marker Specifies the marker that you are defining. Not available for search types that deploy specific markers.

Search Domain Defines the area where the marker can move or search. For full span, the marker searches for specified values within the full measurement span. For user span, the marker searches for specified values within a measurement span that you define. [Learn more about Search Domain.](#)

Search Type

Note You must either press **Execute** or check **Tracking** to initiate all search types. If there is no valid data match for the search type, the marker will not move from its current position.

- **Execute** Click to cause the marker to search for the specified criteria.
- **Tracking** Check to cause the marker to search for the specified criteria with each new sweep. The searches begin with the first sweep after Tracking has been checked, based on the current search type and domain information. Therefore, make sure that the search criteria are in the desired state before using the data. You cannot manually change the stimulus setting for a marker if Tracking is selected for that marker.

Maximum Marker locates the maximum (highest) data value.

Minimum Marker locates the minimum (lowest) data value.

Next Peak Marker locates the peak with the next lower amplitude value relative to its starting position.

Peak Right The marker locates the **next valid peak to the right** of its starting position on the X-axis.

Peak Left The marker locates the **next valid peak to the left** of its starting position on the X-axis.

- **Threshold** - Minimum amplitude (dB). To be considered valid, the peak must be **above** the threshold level. The valley on either side can be below the threshold level.
- **Excursion** The vertical distance (dB) between the peak and the valleys on both sides. To be considered a peak, data values must "fall off" from the peak on both sides by the excursion value.

For more information, see [What is a Peak?](#)

Target Enter the Target value. The marker moves to the first occurrence of the Target value to the right of its current position. Subsequent presses of the Execute button cause the marker to move to the next value to the right that meets the Target value. When the marker reaches the upper end of the stimulus range, it will "wrap around" and continue the search from the lower end of the stimulus range (left side of the window).

- If **Discrete Marker** is OFF, the marker locates the interpolated data point that equals the target value.
- If **Discrete Marker** is ON and there are two data points on either side of the target value, the marker locates the data point closest to the Target value

Bandwidth Markers

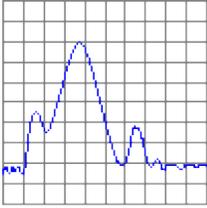


Four markers are automatically deployed to find the first negative or positive bandpass in the selected search domain.

To create Bandwidth markers:

1. Press **SEARCH**, then **[Bandwidth]**
 2. Specify the level in dB from the peak or valley where bandwidth is measured.
- Bandwidth Search can be used **ONLY** with [Log Mag display format](#).
 - To use Bandwidth Search on a peak or valley other than the maximum or minimum values, change the Search Domain.

Enter a **Negative** number to search for a **Peak** bandpass, such as a filter S21 response:



- Marker 1: Maximum value within the [Search Domain](#).
- Marker 2: Specified level DOWN the left of the peak.
- Marker 3: Specified level DOWN the right of the peak.
- Marker 4: Center frequency between markers 2 and 3.

Enter a **Positive** number to search for a **Valley** bandpass, such as a filter S11 response:



- Marker 1: Minimum value within the [Search Domain](#).
- Marker 2: Specified level UP the left of the valley.
- Marker 3: Specified level UP the right of the valley.
- Marker 4: Center frequency between markers 2 and 3.

The following four values are displayed for Bandwidth Search:

- **BW:** (Marker 3 x-axis value) - (Marker 2 x-axis value) = width of the filter.
- **Center:** Mathematical midpoint between markers 2 and 3.
- **Q:** Ratio of Center Frequency to Bandwidth (Center Frequency / Bandwidth).
- **Loss:** Y-axis value of Marker 4. This is the loss of the filter at its center frequency. The ideal filter has no loss (0 dB) in the passband.

Compression Markers



Uses the active marker to find the specified gain **Compression Level**. [Learn more about Gain Compression](#).

Note: Valid ONLY for S21 (Gain) measurements with a [Power Sweep](#).

To create Compression markers:

1. Press **SEARCH**, then **[Compression]**
2. Specify the compression level in dB,
3. Optionally press **Tracking** to search for the specified compression level with each sweep.

Linear gain is defined as the Y-axis value (gain) of the first data point of the [Search Domain](#) (Full Span by default).



Marker > N: X-axis value and Y-axis value

Comp Pin: Input power (marker X-axis value)

Comp Pout: Output power (Pin + gain)

Comp Level: Compression level found.

- When [Discrete](#) is OFF (default setting) the marker finds the exact specified compression, interpolated between the two closest data points and calculates the Comp Pin and Comp Pout value for that point.
- When Discrete is ON (not interpolated), the marker resides on the closest data point to the requested compression level.

Comp. Not Found: Displayed when the requested compression level is not found.

About PSAT and PNOP Markers

Compression measurements based on the Pout vs Pin curves are common in the satellite test industry. In the case of Travelling Wave Tube (TWT) amplifiers, PSAT markers identify the normal operating point near saturation, and the amplifiers are operated with the power slightly backed-off approximately 0.03 to 0.1 dB. For TWT amplifiers, the saturation curve always "folds over" and produces a maximum power out.

For Solid State Power Amplifiers (SSPA), the saturation is not as well defined. A common reference is the Normal Operating Point, which is a power backed-off by 8 to 10 dB from the maximum power. In this case, the normal operating point marker replaces the Psat with the PNOP values. Also, because the backoff is important, the backoff output and input powers are displayed (PBO Out), (PBO in) as well as gain at back off (PBO Gain).

Power Saturation (PSAT) Markers

Uses Markers 1, 2, and 3 to quickly identify output power saturation parameters of an amplifier.

Back-off is a point at which the output power is sufficiently lower than the saturated output power so that the device under test behaves in a more linear fashion.

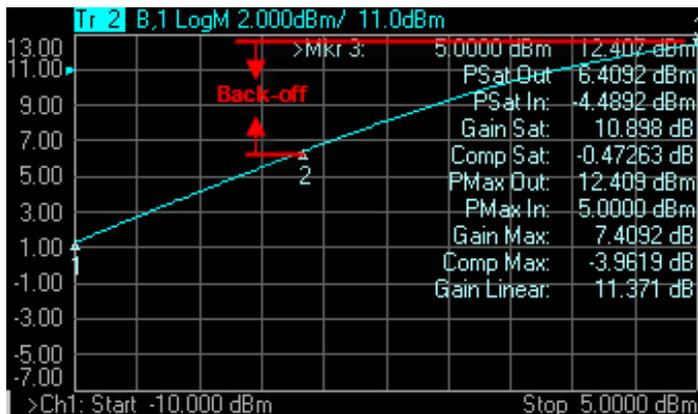
Note: Valid ONLY for Power IN vs Power OUT measurements. These markers can also be used on a CompOut trace with [Compression Analysis](#) mode in the Gain Compression Application.

To make a Power IN (X-axis) vs Power OUT (Y-axis) measurement:

1. [Preset](#)
2. Set [Sweep Type](#): Power Sweep
3. Set [Trace Meas](#) to "B" Receiver
4. Connect DUT input to port 1
5. Connect DUT output to port 2

To create PSAT markers:

1. Press **SEARCH**, then **[Search...]**
2. From Search Type, select **Power Saturation**
3. For **PMax Back-Off**, enter the Y-axis (Power OUT) difference between the Max Power marker (3) and the Back-off marker (2).
4. Press **Execute** or check **Tracking**. [Learn more](#).



This setting uses **three** markers to calculate and display 10 values.

The three markers:

- Marker 1: Linear gain; the first data point in the sweep.
- Marker 2: Specified output power **Back-off** from max power.
- Marker 3: Max Power output; usually the last data point.

The 10 displayed values:

Param	Description	Calculated from...
>Mkr 3	Marker 3 X-axis and Y-axis value	Marker 3 X-axis and Y-axis value
PSat Out	Output power at the saturation point.	Marker 2 Y-axis value
PSat In	Input power at the saturation point.	Marker 2 X-axis value
Gain Sat	Gain at the saturation point.	Psat Out - Psat In
Comp Sat	Compression at the saturation point.	Gain Sat - Gain Linear
PMax Out	Maximum output power.	Marker 3 Y-axis value
PMax In	Input power at the maximum output power.	Marker 3 X-axis value
Gain Max	Gain at the maximum output power.	PMax Out - PMax In
Comp Max	Compression at the maximum output power.	Gain Max - Gain Linear
Gain Linear	Linear gain at the first data point.	Marker 1 - Y-axis value MINUS X-axis value

- **Comp. Not Found** is displayed when the requested Back-off point is not found.
- When [Discrete](#) marker is NOT selected (the default setting), the three markers find an interpolated value between the two closest data points.
- When [Discrete](#) marker is selected (NOT interpolated), the three markers reside on the closest data points.

Programming Commands

Power Normal Operating Point Marker

Uses Markers 1, 2, 3, and 4 to quickly identify Normal Operating Point parameters of an amplifier.

Back-off is a point at which the output power is sufficiently lower than the saturated output power

so that the device under test behaves in a more linear fashion.

The power normal operating point is the output power where the input is offset from the back-off input power by the Pin Offset.

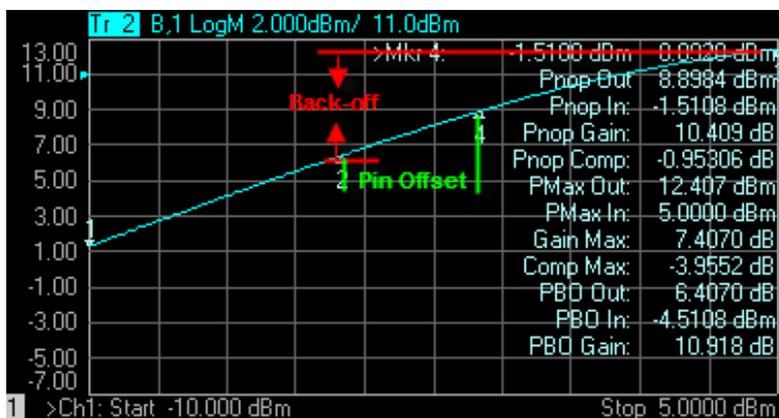
Note: Valid ONLY for Power IN vs Power OUT measurements.

These markers can also be used on a CompOut trace with [Compression Analysis](#) mode in the Gain Compression Application.

See [Power Saturation](#) to learn how to make a Power IN (X-axis) vs Power OUT (Y-axis) measurement.

To create PSAT markers:

1. Press **SEARCH**, then [**Search...**]
2. From Search Type, select **Normal Operating Pt**
3. For **Back-Off**, enter the Y-axis (Power OUT) difference between the Max Power marker (3) and the Back-off marker (2).
4. For **Pin Offset**, enter the X-axis (Power IN) difference between Back-off marker (2) and PNOP marker (4).
5. Press **Execute** or check **Tracking**. [Learn more](#).



This setting uses **four** markers to calculate and display 12 values.

The four markers:

- Marker 1: Linear gain; the first data point in the sweep.
- Marker 2: Max Output Power MINUS the specified Output (Y-axis) **Back-off** value in dB.
- Marker 3: Max Output Power; usually the last data point in the sweep.

- Marker 4: X-axis value of Back-off (Marker 2) plus the **Pin Offset** (X-axis) value in dB.

The 12 displayed values:

Param	Description	Calculated from...
>Mkr 4	Marker 4 - X-axis and Y-axis values	Marker 4 - X-axis and Y-axis values
Pnop Out	Output power at the power normal operating point.	Marker 4 Y-axis value
Pnop In	Input power at the power normal operating point.	Marker 4 X-axis value
Pnop Gain	Gain at the power normal operating point.	Pnop Out - Pnop In
Pnop Comp	Compression at the power normal operating point.	Pnop Gain - Linear Gain*
PMax Out	Maximum output power.	Marker 3 Y-axis value
PMax In	Input power at the maximum output power.	Marker 3 X-axis value
Gain Max	Gain at the maximum output power.	PMax Out - PMax In
Comp Max	Compression at the maximum output power.	Gain Max - Linear Gain*
PBO Out	Output power at the back-off point.	Marker 2 Y-axis
PBO In	Input power at the back-off point.	Marker 2 X-axis
PBO Gain	Gain at the back-off point.	PBO Out - PBO In

***Linear Gain (not shown):** Marker 1 - Y-axis value MINUS X-axis value

- **PNOP Not Found** is displayed when the requested back-off level is not found.
- When [Discrete](#) marker is NOT selected (the default setting), the four markers each find an interpolated value between the two closest data points.
- When [Discrete](#) marker is selected (NOT interpolated), the four markers each reside on the closest data point.

What Is a "Peak"?

You define what the analyzer considers a "peak" by selecting the following two peak criteria settings:

- **Threshold** - Minimum amplitude (dB). To be considered valid, the peak must be **above** the threshold level. The valley on either side can be below the threshold level.
- **Excursion** - The vertical distance (dB) between the peak and the valleys on both sides. To be considered a peak, data values must "fall off" from the peak on both sides by the excursion value.

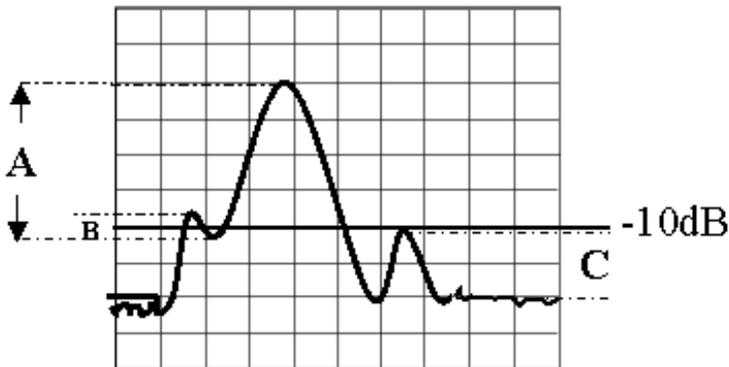
Example:

Threshold Setting: -10dB

Excursion Setting: 1dB

Scale = 1 dB / Division

Mouse over the graphic to find a valid peak.



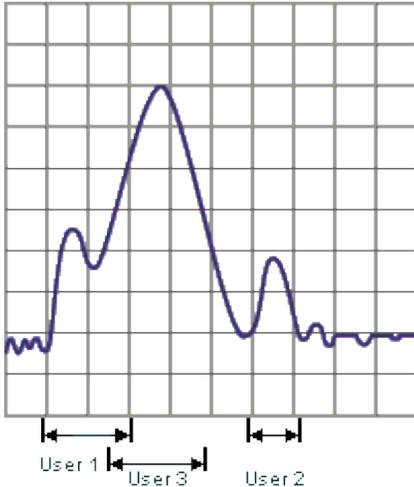
- **Peak A** = Valid Peak (Above Threshold and Excursion Settings)
- **Peak B** = Invalid Peak (Below Excursion Setting)
- **Peak C** = Invalid Peak (Below Threshold Setting)

Search Domain

Search domain settings restrict the stimulus values (X-axis for rectangular format) to a specified span. Set the Start and Stop stimulus settings of these **User** spans. If Start is greater than Stop, the marker will not move. [Learn how to set Search Domain.](#)

- The default domain of each new marker is "full span".
- There are 16 user-defined domains for every channel.
- The user-defined domains can overlap.
- More than one marker can use a defined domain.
- Search Domain settings are shared with [Trace Statistics User Ranges](#)

The graphic below shows examples of search domains.



Search Within

Using the [Zoom feature](#), you can search for the Maximum, Minimum, or a Target value within a 'zoomed' frequency range. The zoomed frequency range becomes the **User 16** Search Domain span.

A marker is created if not already present on the trace. If markers are already present on the trace, the lowest marker is moved to the found value.

1. Left-click the mouse or use a finger, then drag across a portion of a trace.
2. Release the mouse or lift the finger.
3. Select **Search Within**.
4. Then choose from the following:
 - **Max** - A marker moves to the HIGHEST value within the zoomed range.
 - **Min** - A marker moves to the LOWEST value within the zoomed range.
 - **Target** - A marker moves to the first value within the zoomed range that is currently set in the [Marker Search 'Target' setting](#). The same Discrete Marker rules apply as those for the standard Target Marker Search.

Marker Functions - Change Instrument Settings

The following settings change the relevant PNA settings to the position of the active maker.

How to change instrument settings using markers

Using front-panel HARDKEY [softkey] buttons

1. Press **MARKER**
2. then [**Marker Function**]

Using a mouse with PNA Menus

1. Click **Marker/Analysis**
2. then **Marker Function**

Programming Commands

Marker Function dialog box help



Note: Marker Functions do not work with channels that are in [CW](#) or [Segment Sweep](#) mode.

Marker =>Start Sets the start sweep setting to the value of the active marker.

Marker =>Stop Sets the stop sweep setting to the value of the active marker.

Marker =>Center Sets the center of the sweep to the value of the active marker.

Marker =>Ref Level Sets the screen [reference level](#) to the value of the active marker.

Marker =>Delay The phase slope at the **active marker** stimulus position is used to adjust the line length to the receiver input. This effectively flattens the phase trace around the active marker. Additional Electrical Delay adjustments are required on devices without constant group delay over the measured frequency span. You can use this to measure the electrical length or deviation from linear phase.

This feature adds phase delay to a variation in phase versus frequency; therefore, it is only applicable for ratioed measurements. See [Measurement Parameters](#).

Marker =>Span Sets the sweep span to the span that is defined by the [delta marker](#) and the marker that it references. Unavailable if there is no delta marker.

Marker =>CW Freq Sets the CW frequency to the frequency of the active marker. NOT available

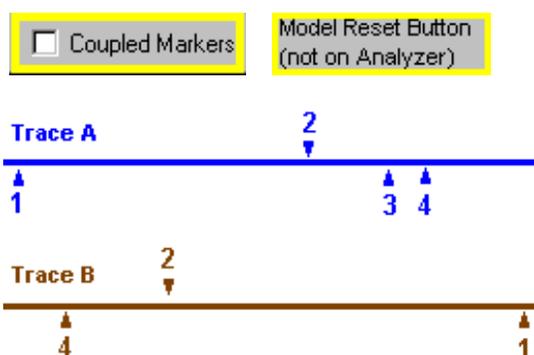
when the channel is in CW or Power Sweep. Use this function to first set the CW Frequency to a value that is known to be within the current calibrated range, THEN set [Sweep Type](#) to Power or CW.

Coupled Markers

The coupled markers feature causes markers on different traces to line up with the markers on the selected trace. Markers are coupled by marker number, 1 to 1, 2 to 2, 3 to 3, and so forth. If the x-axis domain is the same (such as frequency or time), coupling occurs across all channels, windows, and traces. Trace markers in a different x-axis domain will not be coupled. If a trace marker has no marker to couple with on the selected trace, the marker remains independent.

Coupled Markers Model

This model simulates the use of coupled markers in the PNA:



1. Click **Trace A** or **Trace B**
2. Click **Coupled Markers**
3. Notice the following:
 - Markers on the unselected trace move to the x-axis position of the selected trace.
 - If a marker number on the unselected trace has no corresponding marker on the selected trace, no movement occurs for that marker.
4. Click **Reset** to run the model again. There is no Reset for coupled markers on the PNA.

Marker Display

The marker display dialog allows you to change how markers and the associated readout is displayed on the PNA screen. Several marker display features also apply to [Statistics](#) display.

How to change marker display settings

Right-click on a marker readout, then click **Marker Display** or:

Using front-panel

HARDKEY [softkey] buttons

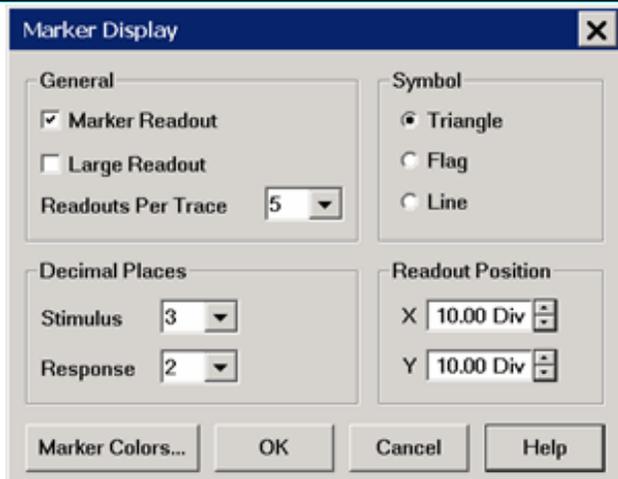
1. Press **MARKER**
2. then **[Properties]**
3. then **[Marker Display]**

Using a mouse with PNA Menus

1. Click **Marker/Analysis**
2. then **Marker**
3. then **Marker Display**

Programming Commands

Marker Display dialog box help



The following settings apply to readouts of ALL currently-displayed marker, bandwidth, and [trace statistics](#).

These settings revert to their defaults on Preset but ARE stored with [Instrument State](#) and [User Preset](#).

Marker Readout

Checked - Shows readout information.

Cleared - Shows NO readout information.

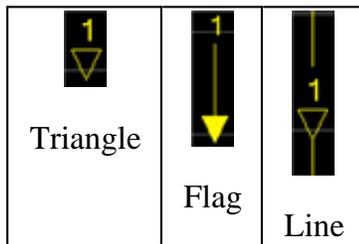
Large Readout

Checked - Shows the marker readout in large font size for easy reading. However, all readout lines may not be visible.

Cleared - Shows the marker readout in normal font size.

Readouts Per Trace Choose the quantity of marker readouts to show in the window for each trace. Choose to display up to 10 readouts per trace, up to 20 readouts per window. When more markers are present than the specified quantity of readouts, the marker numbers for which readouts are displayed can change depending on the marker number that is active. Readouts Per Trace can be set independently for each window.

Symbol - Choose from the following marker symbols.



Line symbols are NOT used on Smith or Polar [display formats](#).

Symbols can be set independently for each window.

Decimal Places Choose the marker readout resolution to display. These values also apply to the readouts that are displayed in the [marker table](#). Decimal Places can be set independently for each window.

Stimulus (X-axis) - Choose from **2** to **6** places after the decimal point. Default is 3.

Response (Y-axis) - choose from **1** to **4** places after the decimal point. Default is 2.

Readout Position Choose where to place the marker readouts. Marker readouts are right-justified on the specified X-axis and Y-axis position. The default position (10.0, 10.0) is the upper-right corner of the grid. Position (1.0,1.0) is the lower-left corner. Readout position can be set independently for each window.

Note: Readout Position can also be changed using a mouse by left-clicking on the top readout and dragging to the new position.

Marker Colors Starts the Display Colors dialog with only the marker colors available. [Learn more](#).

Marker Table

You can display a table that provides a summary of marker data for the active trace. The marker data is displayed in the specified format for each marker.

How to view the Marker Table

Using front-panel HARDKEY [softkey] buttons

1. Press **DISPLAY**
2. then **[More]**
3. then **[Tables]**
4. then **[Marker Table]**

Using a mouse with PNA Menus

1. Click **Response**
2. then **Display**
3. then **Tables**
4. then **Marker Table**

Programming Commands

Last Modified:

18-Jul-2013	Added Search Within
29-Mar-2013	Added marker formats
3-May-2012	Removed c models
7-Sep-2011	Edits to PSAT and PNOP
7-Jun-2011	Linked Search domain to User Range
5-Aug-2010	Added marker display
9-Feb-2010	Added PSAT, PNOP, and new dialog
6-Feb-2009	Added compression marker
3-Oct-2008	Added Marker->CW function
3-Sep-2008	Removed legacy content
4-Jan-2008	Added bookmark to move marker
17-Jul-2007	Clarified bandwidth search
2-Feb-2007	MX Added UI

Using Math / Memory Operations

You can perform four types of math on the active trace versus a memory trace. In addition three statistics (Mean, Standard Deviation and Peak to Peak) can be calculated and displayed for the active data trace.

- [Trace Math](#)
- [Trace Statistics](#)

Note: Trace Math (described here) allows you to quickly apply one of four math operations using memory traces. [Equation Editor](#) allows you to build custom equations using several types of traces from the same, or different channels.

Other Analyze Data topics

Trace Math

To perform any of the math operations, you must first store a trace to memory. You can display the memory trace using the [View](#) options.

Trace math is performed on the complex data before it is formatted for display. See the [PNA data processing map](#).

Markers can be used while viewing a memory trace.

How to select Trace Math

Using front-panel HARDKEY [softkey] buttons

1. Press **MEMORY**

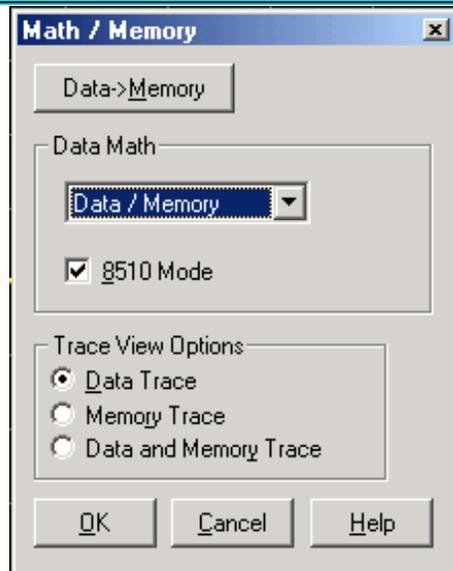
Using a mouse with PNA Menus

1. Click **Marker/Analysis**
2. then **Memory**
3. then **Memory**

Normalize, available only from the Memory menu, (not on the Math / Memory dialog), performs the same function as **Data=>Memory**, then **Data / Memory**.

[Programming Commands](#)

Math / Memory dialog box help



Normalize, available only from the Memory menu, (not on the Math / Memory dialog), performs the same function as **Data=>Memory**, then **Data / Memory**.

Data=>Memory Puts the active data trace into memory. You can store one memory trace for every displayed trace.

Note: Many PNA features are NOT allowed on Memory traces. For example, Memory traces can NOT be saved to any [file type](#) (PRN, SNP, CTI, CSV, MDF). However, you can restore a memory trace to a data trace using the **Memory-to-Data** utility at the <http://na.tm.agilent.com/pna/apps/applications.htm> website.

Data Math

All math operations are performed on linear (real and imaginary) data before being formatted. See the PNA Data flow (below).

Data (or OFF) Does no mathematical operation.

Data / Memory - Current measurement data is divided by the data in memory. Use for ratio comparison of two traces, such as measurements of gain or attenuation. [Learn more](#).

Data – Memory - Data in memory is subtracted from the current measurement data. For example, you can use this feature for storing a measured vector error, then subtracting this error from the DUT measurement. [Learn more](#).

Data + Memory - Current measurement data is added to the data in memory. [Learn more](#).

Data * Memory - Current measurement data is multiplied by the data in memory. [Learn more](#).

8510 Mode - [Learn more](#).

Trace View Options

Data Trace Displays ONLY the Data trace (with selected math operation applied).

Memory Trace Displays ONLY the trace that was put in memory.

Data and Memory Trace Displays BOTH the Data trace (with selected math operation applied), and the trace that was put in memory.

[Learn more about Trace Math](#) (scroll up)

(Data / Memory) and (Data - Memory)

(Data / Memory) and (Data - Memory) math operations are performed on linear data before it is formatted. Because data is often viewed in log format, it is not always clear which of the two math operations should be used. Remember: dividing linear data is the same as subtracting logarithmic data. The following illustrates, in general, when to use each operation.

Use **Data / Memory** for normalization purposes, such as when comparing S21 traces "before" and "after" a change is made or measurement of trace noise. In the following table, the Data/Mem values intuitively show the differences between traces. It is not obvious what Data-Mem is displaying.

S21 values to compare	Data/Mem	Data-Mem
0.5 dB and 0.6 dB	0.1 dB	-39 dB
0.5 dB and 0.7 dB	0.2 dB	-33 dB

Use **Data - Memory** to show the relative differences between two signals. Use for comparison of very small signals, such as the S11 match of two connectors.

In the following table, Data/Mem shows both pairs of connectors to have the same 2 dB difference. However, the second pair of connectors have much better S11 performance (-50 and -52) and the relative significance is shown in the Data-Mem values.

S11 values to compare	Data/Mem	Data-Mem
-10 dB and -12 dB	2 dB	-24 dB
-50 dB and -52 dB	2 dB	-64 dB

Data * Memory and Data + Memory

Use **Data * Memory** and **Data + Memory** to perform math on an active data trace using data from your own formulas or algorithms rather than data from a measurement. For example, if you want to

simulate the gain of a theoretical amplifier placed in series before the DUT, you could do the following:

1. Create an algorithm that would characterize the frequency response of the theoretical amplifier.
2. Enter complex data pairs that correspond to the number of data points for your data trace.
3. Load the data pairs into memory with SCPI or COM commands. The analyzer maps the complex pairs to correspond to the stimulus values at the actual measurement points.
4. Use the **data + memory** or **data * memory** function to add or multiply the frequency response data to the measured data from the active data trace.

Note: The data trace must be configured before you attempt to load the memory.

Trace Statistics

You can calculate and display statistics for the active data trace. These statistics are:

- Mean
- Standard deviation
- Peak-to-peak values

You can calculate statistics for the full stimulus span or for part of it by using User Ranges.

You can define up to 16 user ranges per channel. These user ranges are the same as the [Search Domain](#) specified for a marker search in that same channel. They use the same memory registers and thus share the same stimulus spans.

The user ranges for a channel can overlap each other.

A convenient use for trace statistics is to find the peak-to-peak value of passband ripple without searching separately for the minimum and maximum values.

The trace statistics are calculated based on the format used to display the data.

- [Rectangular data formats](#) are calculated from the scalar data represented in the display
- [Polar](#) or [Smith Chart](#) formats are calculated from the data as it would be displayed in [Log Mag](#) format

[See how to make Trace Statistics display settings.](#)

How to activate Trace Statistics

Using front-panel HARDKEY [softkey] buttons

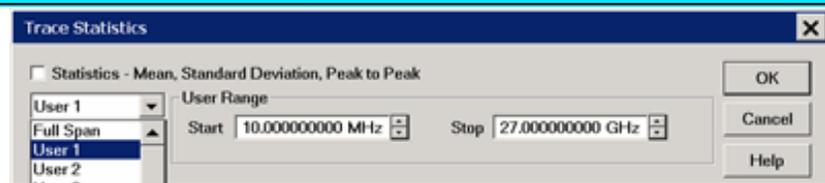
1. Press **ANALYSIS**
2. then **[Statistics]**
3. then **[Trace Statistics]**

Using a mouse with PNA Menus

1. Click **Marker/Analysis**
2. then **Analysis**
3. then **Trace Statistics**

Programming Commands

Trace Statistics dialog box help



[See how to make Trace Statistics display settings.](#)

Statistics Check to display mean, standard deviation, and peak to peak values for the active trace.

Span Specifies the span of the active trace where data is collected for a math operation. You can select Full Span, or define up to 16 user spans per channel with Start and Stop. You can also define the user spans from the Search Domain selector on the [Marker Search dialog box](#).

Start Defines the start of a user span.

Stop Defines the stop of a user span.

[Learn more about Trace Statistics](#) (scroll up)

Last Modified:

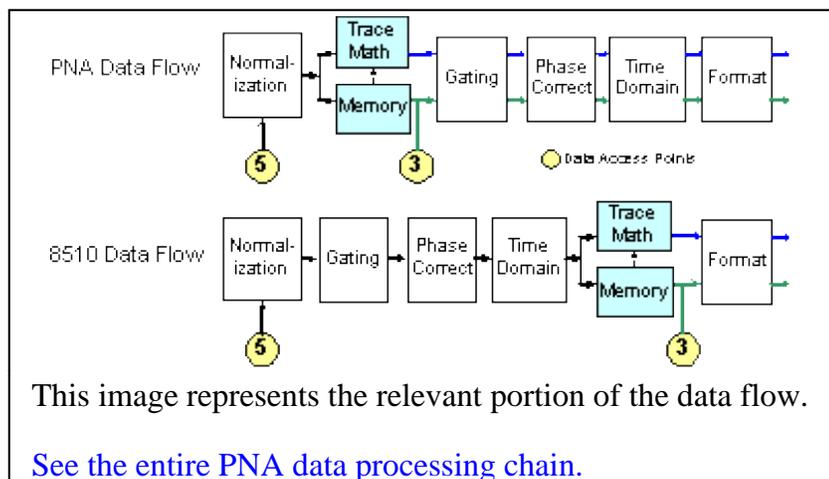
- | | |
|-------------|-----------------------------------|
| 7-Jun-2011 | Moved 8510 mode to separate topic |
| 3-Sep-2008 | Removed legacy content |
| 28-Aug-2008 | Added Memory trace utility note |
| 22-Apr-2008 | Added 8510 preference link |
| 27-Aug-2007 | Edited trace display settings |
| 2-Feb-2007 | MX added UI |

8510 Mode

On the [Trace Math](#) dialog, check 8510 Mode to simulate the Agilent 8510 data processing chain as it pertains to Trace Math and Memory. This setting applies to all channels. When the box is checked or cleared, the PNA performs an [Instrument Preset](#) and retains its setting through subsequent Instrument Presets.

This setting can be saved as part of an [instrument state](#). However, when recalled, this setting is assumed only temporarily. When a subsequent PNA Preset is performed, the PNA reverts to the setting that was in effect before the state was recalled.

You can [set a preference](#) to always use 8510 mode.



A settings change in any of the operations that occur after the Memory operation on the above **PNA Data Flow** diagram changes both the Data trace and the Memory trace. For example, after storing a data trace to memory, when you change the format for the Data Trace, the format for the Memory Trace is also changed to the same setting.

Last modified:

12-Apr-2011 New topic- moved from Trace Math and include special case note.

Equation Editor

Equation Editor allows you to enter an algebraic equation that can mathematically manipulate measured data. The results are displayed as a data trace. Data that is used in the equation can be from the same or different channels.

Note: Equation Editor is now available with ALL PNA Apps.

- [Overview](#)
- [How to start Equation Editor](#)
- [Using Equation Editor](#)
- [Data that is used in Equation Editor](#)
- [Trace Settings, Error Correction, and an Example](#)
- [Functions and Constants](#)
- [Operators](#)
- [Example Equations](#)
- [Saving Equation Editor Data](#)

See Also

[Using Noise Power Traces in Equation Editor](#)

[Equation Editor and MATLAB](#)

[Equation Editor Import Functions](#)

- [BestFit.dll](#)
- [EqnErrorTerms.dll](#)
- [Expansion.dll](#)

Other 'Analyze Data' topics

Overview

Equation Editor allows you to enter an algebraic equation of standard mathematical operators and functions, referencing data that is available in the PNA. Once a valid equation is entered and enabled,

the display of the active trace is replaced with the results of the equation, and updated in real-time as new data is acquired. For equations that can be expressed with Equation Editor's supported functions, operators, and data, there is no need for off-line processing in a separate program.

For example, enter the equation $S_{21} / (1 - S_{11})$. The resulting trace is computed as each S_{21} data point divided by one minus the corresponding S_{11} data point. For a 201 point sweep setup, the computation is repeated 201 times, once for each point.

As another example, suppose you want the PNA to make a directivity measurement of your 3-port DUT. This is not a native PNA measurement, but can be achieved using the Equation Editor. The desired result is the sum and difference of LogMag formatted traces, expressed as: $S_{12} + S_{23} - S_{13}$.

Because Equation Editor operates on **unformatted complex data**, the required equation is:

```
DIR = S12 * S23 / S13
```

DIR becomes a display label to help you identify the computed data trace.

On the equation trace, set the format to LogMag.

How to start Equation Editor

Using front-panel HARDKEY [softkey] buttons

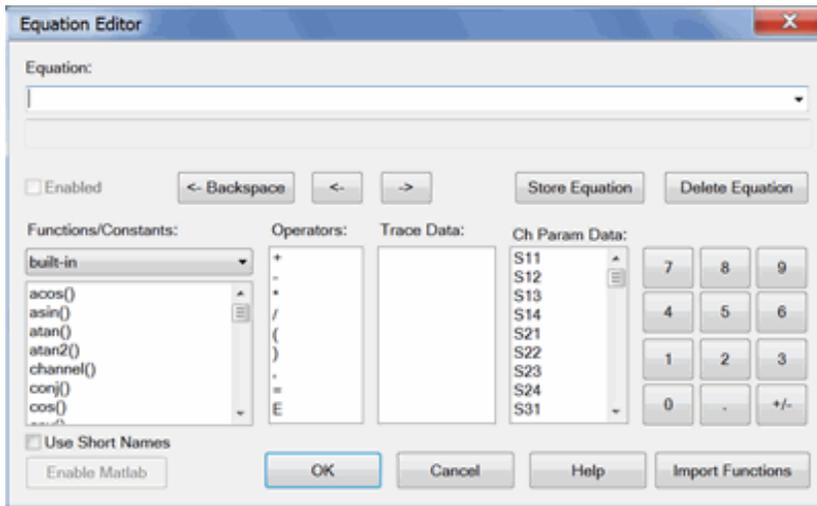
1. Press **ANALYSIS**
2. then **[Equation Editor]**

Using a mouse with PNA Menus

1. Click **Marker/Analysis**
2. then **Analysis**
3. then **Equation Editor**

◀ Programming Commands ▶

Equation Editor dialog box help



Notes

- **Double-click**, or type, the Functions, Operators, and Data to build an Equation.
- Scroll down to learn more about [Using Equation Editor](#)

Equation: The field in which equations are built. Click the down arrow to the right to use or modify equations that have been previously saved. This is where equations are saved when you press 'Store Equation'.

Enabled Check this box to enable the equation that is currently in the Equation field. If the Enabled box is not available, then the equation is not valid. If a data trace is used that is from a different channel than the Equation trace, the channels **MUST** have the same number of data points to be valid.

<-Backspace Moves the cursor to the left while erasing characters.

<- Moves the cursor to the left without erasing characters.

-> Moves the cursor to the right without erasing characters.

Store Equation Press to save the current equation. To later recall the equation, click the down arrow to the right of the equation.

Delete Equation Removes the current equation from the drop-down list.

Functions/Constants: See [descriptions of Functions](#).

Select the "library" of functions to view. The "built-in" library appears by default which includes the standard functions of equation editor. Other functions that can appear here are functions that you have written and imported. [Learn more.](#)

Operators: See [descriptions of Operators](#).

Trace Data: Select from ALL of the currently **displayed** traces on ALL channels.

Ch Param Data: Select from **undisplayed** data that is available **ONLY** from the active channel (same channel as the equation trace). See [Data that is used in Equations](#).

Note: With an external test set enabled, only parameters involving ports 1 through 4 are listed. However, all available parameters can be typed directly into the **Equation** field.

Keypad: Provided to allow navigation of the entire dialog with a mouse.

Import Functions Click to launch the [Import Functions](#) Dialog box.

Use Short Names Some functions have shortened names that are entered automatically when checked. Both long and short names can be used interchangeably.

Enable Matlab Available when a full MATLAB version is installed by you on the PNA. [Learn more](#).

Using Equation Editor

1. Pick a trace in which to enter the equation

- Equation Editor works on the active trace.
- Either create a new trace, or click the [Trace Status](#) button on an existing trace to make the trace active.

2. Enter an equation

Start Equation Editor [See how](#).

- The equation text can be in the form of an expression $(S21)/(1-S11)$ or an equation $(DIR = S12 * S23 / S13)$. This topic refers to both types as equations.
- Either type, or double-click the Functions, Operators, and Data to build an equation.
- Functions and Constants ARE case-sensitive; Data names are NOT case sensitive.
- [Learn more about referring to data traces](#).

3. Check for a valid equation

When a valid equation is entered, the Enabled checkbox becomes available for checking. When the Enabled box is checked:

- The Equation Trace becomes computed data.
- The equation is visible on the [Trace Status](#) (up to about 10 characters).
- The equation is visible in the trace [Title](#) area (up to about 45 characters) when the Equation trace is active.
- The equation is visible in the [Status Bar](#) at the bottom of the display. This is updated only after the equation is entered and the [Trace Status](#) button is clicked.
- If an equation is NOT valid, and a trace from a different channel is used, make sure the number of data points is the same for both channels.

Learn more about the [Functions](#), [Operators](#), and [Data](#) that are used in Equation Editor.

Data that is used in Equation Editor

Definitions

- **Equation trace** A trace in which an equation resides.
- **Referred trace** A trace that is used as data in an equation.

Example: $eq=Tr2+S11$ is entered into **Tr1**.

Tr1 becomes an equation trace.

Tr2 and **S11** are both referred traces because they are used in the equation trace.

Notes

- Referred traces are processed one data point at a time. For example, the expression $S11/S21$ means that for each data point in S11 and S21, divide point N of S11 by point N of S21.
- Once an equation is enabled, the trace is no longer identified by its original measurement parameter. It becomes an equation trace.
- An equation trace can NOT refer to itself. For example, an equation in Tr1 cannot refer to trace Tr1.
- Referred traces can be selected from S-Parameters, Receiver data, and [Memory traces](#).
- [See note regarding External Test Sets.](#)
- See [Using Noise Power Traces in Equation Editor](#)

There are three ways to refer to traces:

The following distinction is important when discussing the three ways to refer to traces/data.

- **Trace** - a sequential collection of data points that are displayed on the PNA screen.

- **Data** - PNA measurements that are acquired but not displayed. When an equation trace refers to data that is not displayed, the PNA will automatically acquire the data.

1. Using **TrX** Trace notation (for example, Tr2).

When a trace is created, check "[Show Tr Annotation](#)" to see the **Tr** number of that trace.

- **Simple** - ALWAYS refers to displayed traces.
- Must be used for referring to traces in a different channel as the equation trace.
- All [trace settings](#) are preserved in the equation trace. If you do NOT want a trace setting to be used in the equation trace, you must disable it in the referred trace.
- If the referred trace is error corrected, then that data is corrected in the equation trace.
- Used to refer to a memory trace (it must already be stored in memory). Append .MEM to the **TrX** trace identifier. For example, **Tr2.mem** refers to the memory trace that is stored for Tr2.

2. Using **S-parameter** notation (for example, S11/S21)

- **Convenient** - ALWAYS refers to data that is NOT displayed.
- Refers to data that resides in the same channel as the equation.
- NOT the same as referring to a displayed S11 trace using **TrX** notation. [See Example](#).
 - The referred data includes NO [trace settings](#).
 - If the channel has error correction available, then it can be applied by turning error correction ON for the Equation trace.

3. Using **Receiver** notation (for example AB_2); NOT case sensitive.

At least one receiver is required, followed by an underscore and a number.

- The **letters** before the underscore refer to the receivers.
 - Letters alone refer to physical receivers.
 - Letters immediately followed by numbers refer to logical receivers. [Learn more](#).
 - If two receivers are referenced, they are ratioed.
- The **number** after the underscore refers to the source port for the measurement.

Examples

- AR1_2 = physical receiver A / physical receiver R1 with 2 as the source port.
- a3b4_1 = reference receiver for port 3 / test port receiver for port 4 with 1 as the source port.

[Learn more about ratioed and unratioed receiver measurements.](#)

Receiver notation is like S-parameter notation in that:

- Refers to data that is NOT displayed and resides in the same channel as the equation.
- The referred data includes NO trace settings.
- If the channel has error correction available for that receiver, then it can be applied by turning error correction ON for the Equation trace.

Referring to Traces in a different channel

When the equation trace refers to a trace on a different channel:

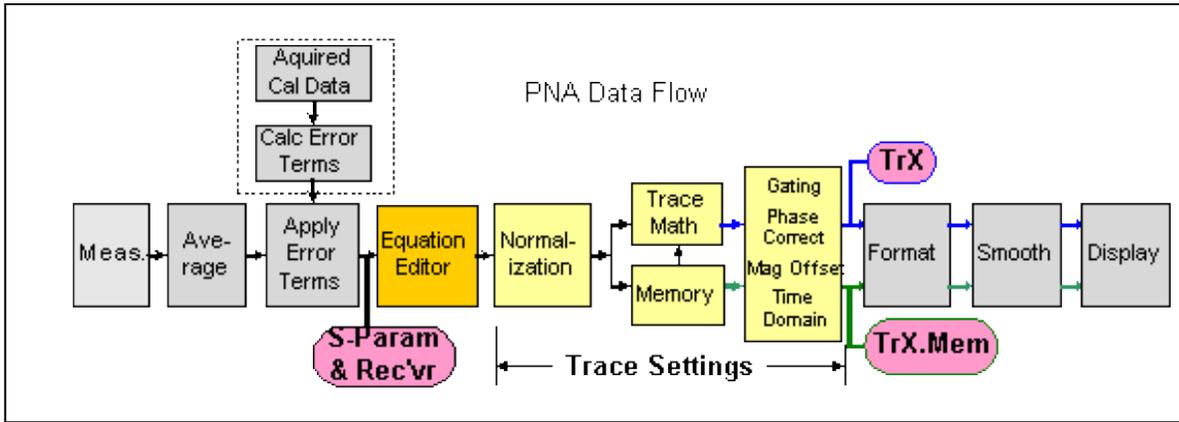
- The trace must already be displayed.
- Must refer to the trace using **TrX** notation.
- The Equation trace and the referred trace MUST have the same number of data points or the Enable checkbox will not be available.
- The Equation trace is updated when the last referred data in the same channel is acquired. Therefore, to prevent 'stale' data from being used, the Equation trace must be on a higher numbered channel than the referred trace. This is because the PNA acquires data in ascending channel number order - first channel 1, then channel 2, and so forth. If the Equation trace is on channel 1, and it refers to a trace on channel 2, the Equation trace will update after channel 1 is finished sweeping, using 'old' data for the channel 2 trace.

Port Extensions and Equation Editor

When using port extension with an equation, turn Fixturing ON to ensure that the underlying parameters have port extension properly applied. Learn more.

Trace Settings, Error Correction, and an Example

This discussion highlights the differences between using **S-parameter / Receiver** notation and **TrX** notation when referring to traces. The key to understanding the differences is realizing that **S-parameter / Receiver** notation ALWAYS refers to data that is NOT displayed.



- **Trace Settings** Normalization, Trace Math, Gating, Phase and Mag Offset, Electrical Delay, Time Domain.
- **Equation Editor** processing occurs on the **equation trace** immediately after error correction.
- **Referred Data/Trace** (used in the equation) is taken from the following locations:
 - When using **TrX** notation, data is taken immediately before formatting . These traces are always displayed and include **Trace Settings**.
 - When using **S-parameter / Receiver** notation, data is taken immediately after error correction. This data is NOT displayed and includes **NO** trace settings ([see example](#)).

See Equation Editor Notes at [GetData Method](#) or [GetDataByString Method](#).

Error-correction and Equation Editor

Using **TrX** notation:

- The Trace Settings and Error-correction on the referred trace are used in the Equation trace.
- If error correction is NOT ON, then the raw, uncorrected data is used in the equation trace.
- To see if error correction is ON, make the trace active, then see the [Correction level in the status bar](#).
- Turning error correction ON/OFF on the equation trace has no meaning. The referred data that is used in the equation is ALWAYS what determines its level of correction.

Using **S-parameter** and **Receiver** notation:

- Because the data is not displayed, **NO** trace settings are used in the Equation trace.
- Correction can be turned ON/OFF if corrected data is available for the referred data. Exception: When using S-parameter and Receiver notation to refer to a trace on a channel that has been calibrated with a [Response Cal](#) or Receiver Cal, correction can NOT be turned ON, even though the Status Bar indicates otherwise. For example: Tr1 is an S11 measurement with a Response Cal. Tr2 is an equation trace that refers to S11. The Tr2 equation trace is NOT corrected, even though the Status Bar may indicate that it is corrected. However, if Tr2 refers to Tr1 (not S11), the Tr2 equation trace is corrected.

Example

This example illustrates the differences when referring to a trace using **S-parameter** notation and **TrX** notation:



- **Tr1** is an S11 measurement with no equation, 2-port correction ON, and Time Domain transform ON.
- **Tr2** is an equation trace that refers to **Tr1**. Tr2 is corrected because Tr1 is corrected. Tr2 is transformed because Tr1 is transformed. If transform is turned ON for Tr2, the data will be transformed AGAIN, which results in "unusual" data.
- **Tr3** is an equation trace that refers to **S11**. This is NOT the same as referring to Tr1. The S11 trace that is referred to is a different instance of S11 that is NOT displayed, and has NO trace settings. Notice that Tr3 data is NOT transformed, although Tr1 is transformed. Correction for **Tr3** can be turned ON and OFF because a calibration was performed on the channel in which the S11 trace resides.
- **Note:** X- axis annotation of the Equation trace is completely independent of the data that is presented. ONLY the **data values** from a referred trace are used. For example, notice that the Equation trace **Tr2** has Frequency on the X-axis although the referred trace **Tr1** is presented in Time.

Functions and Constants used in Equation Editor

ALL trace data that is used in Equation Editor is unformatted, complex data.

When using a mouse with the PNA, hover over a function in the dialog to learn how it is used.

In the following table,

- Function(scalar x) means that an automatic conversion from a complex number to its scalar magnitude is performed before passing the value to the function.
- Function(complex x) means that the entire complex value is used.
- **a, b, c, d** are arguments that are used in the function.

Function/Constant	Description
acos(scalar a)	returns the arc cosine of a in radians
asin(scalar a)	returns the arc sine of a in radians
atan(scalar a)	returns the arc tangent of a in radians
atan2	returns the phase of complex a = (re,im) in radians has the following two argument sets: <ul style="list-style-type: none"> • atan2(complex a) - returns the phase in radians • atan2(scalar a, scalar b)
conj(complex a)	takes a and returns the complex conjugate
cos(complex a)	takes a in radians and returns the cosine
cpx(scalar a, scalar b)	returns a complex value (a+ib) from two scalar values
e	returns the constant $\approx 2.71828\dots$
exp(complex a)	returns the exponential of a
getNumPoints()	returns the number of points for the current sweep
im(complex a)	returns the imag part of a as the scalar part of the result (zeroes the imag part)
kfac(complex a, complex b, complex c, complex d)	k-factor: $k = (1 - a ^2 - d ^2 + a*d-b*c ^2) / (2 * b*c)$
when entered in EE: kfac(S11,S21,S12,S22)	returns a scalar result - the imaginary part of the complex result is always 0
ln(complex a)	returns the natural logarithm of a
log10(complex a)	returns the base 10 logarithm of a

mag(complex a)	returns $\sqrt{\mathbf{a.re}*\mathbf{a.re}+\mathbf{a.im}*\mathbf{a.im}}$
max(complex a, complex b, ...)	returns the complex value that has the largest magnitude of a list of values.
max_hold(complex a)	holds the current maximums of the sweep. Disable the equation to reset. See example
median(complex a, complex b,...)	returns the median of a list of complex values <ul style="list-style-type: none"> • The median is determined by sorting the values by magnitude, and returning the middle one. • If an even number of values is passed, then the smaller of the two middle values is returned.
min(complex a, complex b, ...)	returns the complex value that has the smallest magnitude of a list of values.
min_hold(complex a)	holds the current minimums of the sweep. Disable the equation to reset. See example
mrkx(a)	returns the x-axis value of the active marker on trace a.
mrky(a)	returns the y-axis value of the active marker on trace a.
mu1(complex a, complex b, complex c, complex d) when entered in EE: mu1(S11,S21,S12,S22)	$\mu1 = (1 - \mathbf{a} ^2) / (d - \text{conj}(\mathbf{a}) * (\mathbf{a}*d-\mathbf{b}*c) + \mathbf{b}*c)$
mu2(complex a, complex b, complex c, complex d) when entered in EE: mu1(S11,S21,S12,S22)	$\mu2 = (1 - \mathbf{d} ^2) / (a - \text{conj}(\mathbf{d}) * (\mathbf{a}*d-\mathbf{b}*c) + \mathbf{b}*c)$
for both mu1 and mu2 (Usually written with the Greek character μ)	<ul style="list-style-type: none"> • conj is the complex conjugate. For scalars \mathbf{a} and \mathbf{b}, $\text{conj}(\mathbf{a}+i\mathbf{b}) = (\mathbf{a}-i\mathbf{b})$ • returns a scalar result - the imaginary part of the complex result is always 0
phase(complex a)	returns $\text{atan2}(\mathbf{a})$ in degrees
PI	returns the numeric constant pi (3.141592), which is the ratio of the circumference of a circle to its diameter
pow(complex a,complex b)	returns \mathbf{a} to the power \mathbf{b}
re(complex a)	returns the scalar part of \mathbf{a} (zeroes the imag part)
sin(complex a)	takes \mathbf{a} in radians and returns the sine
sqrt(complex a)	returns the square root of \mathbf{a} , with phase angle in the half-open interval $(-\pi/2, \pi/2]$

tan(complex a)	takes a in radians and returns the tangent
traceDataArray(complex a)	returns the entire set of points from a sweep. Function is intended to be used as an argument in an custom function to allow access for data array processing.
xAxisArray()	returns the x-axis values for the entire sweep.
xAxisIndex()	returns the current index in the sweep.
xAxisValue()	returns the current value of the x-axis index.

Operators used in Equation Editor

Operator	Description
+	Addition
-	Subtraction
*	Multiplication
/	Division
(Open parenthesis
)	Close parenthesis
,	Comma - separator for arguments (as in S11, S22)
=	Equal (optional)
E	Exponent (as in 23.45E6)

Example Equations

The following examples may help you get started with Equation Editor.

Offset each data point in Tr2 from Tr1 by 2dB

Use the function: pow(complex **a**, complex **b**) -- returns **a** to the power **b**.

$$20\log(a) + 2 = 20\log(x)$$

$$\log(a) + 2/20 = \log(x) \text{ // divide all by 20.}$$

$$x = 10^{(\log(a) + 2/20)} \text{ // swap sides and take 10 to the power of both sides}$$

$$x = 10^{\log(a)} * 10^{(2/20)}$$

$$x = a * 10^{(2/20)}$$

The equation is entered into Tr2 as:

$$\text{Offset}=\text{Tr1}*\text{pow}(10, 2/20)$$

To offset by 5 dB

```
Offset=Tr1*pow(10, 5/20).
```

Balanced Match using a 2-port PNA

```
SDD11 = (S11-S21-S12+S22)/2
```

Conversion loss

```
B_1/pow(10, -15/20)
```

- B_1 is a receiver measurement;
- -15 is the input power in dBm

Third-order intercept point (IP3 or TOI)

```
TR1*sqrt(Tr1/Tr3)
```

- Tr1 = input signal power
- Tr3 = intermodulation power (both traces measured with single receivers)

Harmonics in dBc

```
B_1/Tr2
```

- B_1 is tuned to a harmonic frequency
- Tr2 = power at fundamental frequency, measured with B_1 receiver

PAE (Power Added Efficiency)

$P_{out} - P_{in} / P_{dc}$

Type the following equation into a new trace with an unratiod measurement, such as AI1. The data format is REAL:

```
PAE = 100 * (.001*pow(mag(Tr1),2) - (.001*pow(mag(Tr1),2)/pow(mag(Tr2),2)))/(Tr3*Tr4)
```

Where:

- Tr1 - a trace that measures unratiod B receiver.
- Tr2 - a corrected S21 trace (amplifier gain)
- Tr3 - a trace that measures [ADC voltage](#) (AI1) across a sensing resistor.

- Tr4 = an equation trace containing $I_{supp} = (Tr3 / \text{value of sensing resistor})$.

Data is displayed in Real format with units actually being watts.

1-port Insertion Loss

When it is not possible to connect both ends of a cable to the PNA, a 1-port insertion loss measurement can be made. However, the measured loss must be divided by 2 because the result includes the loss going down **and** coming back through the cable. This assumes that the device is terminated with a short or open to reflect all of the power. The 'divide by 2' operation (for dB) is performed as follows using Equation Editor:

- Tr1 - an S11 trace in log mag format.
- Tr2 - an equation trace containing `sqrt(Tr1)`

Max and Min Hold

These two functions allow you to capture and display either the Maximum or Minimum values for each data point over multiple sweeps.

Maxhold(S21) - displays the maximum value for each data point until reset. Reset by disabling, then enabling the equation. This example refers to an S21 trace that is not displayed.

Saving Equation Editor Data

Equation data can be saved to the PNA hard drive in the following formats:

- [Citifile \(.cti\)](#) - Equation data is saved and recalled. The file header indicates the "underlying" s-parameter trace type.
- [PRN](#) - read by Spreadsheet software. Can NOT be recalled by the PNA.
- [CSV](#) - read by Spreadsheet software. Can NOT be recalled by the PNA.
- [MDIF](#) - compatible with Agilent ADS (Advanced Design System). Can NOT be recalled by the PNA.
- [Print to File](#) (bmp, jpg, png) - saves image of PNA screen.

Equation data can NOT be saved in [.SnP file format](#). When attempting to save an Equation trace in .SnP format, the "underlying" S-parameter data is saved; not Equation data.

Last Modified:

12-Dec-2013	Added Matlab enable
26-Feb-2013	Several additions for A.09.90
6-Apr-2011	Replaced 1-port example with JV ex.
5-Nov-2010	Edited TR annotation
20-Sep-2010	Added file types to save data
2-Sep-10	Added link to NF
11-Jun-2009	Added link to GetData
3-Sep-2008	Removed legacy content
22-Aug-2008	Added xAxisArray() Maxhold() and Minhold() function
6-May-2008	Added Import functions
3-Jul-2007	Added PAE and other notes
18-Jun-2007	Added examples

Using Limit Lines

Limit lines allow you to compare measurement data to performance constraints that you define.

- [Overview](#)
- [Create and Edit Limit Lines](#)
- [Display and Test with Limit Lines](#)
- [Testing with Sufficient Data Points](#)

Other Analyze Data topics

Overview

Limit lines are visual representations on the PNA screen of the specified limits for a measurement. You can use limit lines to do the following:

- Give the operator **visual guides** when tuning devices.
- Provide **standard criteria** for meeting device specification.
- Show the **comparison** of data versus specifications.

Limit testing compares the measured data with defined limits, and provides optional **Pass or Fail** information for each measured data point.

You can have up to **100** discrete lines for each measurement trace allowing you to test all aspects of your DUT response.

Limit lines and limit testing are **NOT** available with **Smith Chart** or **Polar** display format. If limit lines are ON and you change to Smith Chart or Polar format, the analyzer will automatically disable the limit lines and limit testing.

Create and Edit Limit Lines

You can create limit lines for all measurement traces. The limit lines are the same color as the measurement trace.

Limit lines are made up of discrete lines with four coordinates:

- BEGIN and END stimulus - X-axis values.

- BEGIN and END response - Y-axis values.

How to create, edit, and test with Limit Lines

All limit line settings are made with the limit table. Use one of the following methods to show the limit table:

Using front-panel HARDKEY [softkey] buttons

1. Press **ANALYSIS**
2. then **[Limits]**
3. then **[Limit Test]**

Using a mouse with PNA Menus

1. Click **Marker/Analysis**
2. then **Analysis**
3. then **Limit Test**

Programming Commands

Limit Table

	TYPE	BEGIN STIMULUS	END STIMULUS	BEGIN RESPONSE	END RESPONSE
1	MIN	1.930000 GHz	1.990000 GHz	-5.000000 dB	-5.000000 dB
2	MAX	1.000000 GHz	1.500000 GHz	-60.000000 dB	-50.000000 dB
3	MAX	2.050000 GHz	3.000000 GHz	-50.000000 dB	-60.000000 dB
4	OFF	0.000000 GHz	0.000000 Hz	0.000000 dB	0.000000 dB

Note: To ADD a limit line to the table, change the last limit line to either MAX or MIN

1. In the **Type** area of the Limit Table, select **MIN** or **MAX** for Limit Line 1.
 - The MIN value will fail measurements BELOW this limit.
 - The MAX value will fail measurements ABOVE this limit.
2. Click **BEGIN STIMULUS** for Limit Segment 1. Enter the desired value.
3. Click **END STIMULUS** for Limit Segment 1. Enter the desired value.
4. Click **BEGIN RESPONSE** for Limit Segment 1. Enter the desired value.
5. Click **END RESPONSE** for Limit Segment 1. Enter the desired value.
6. Repeat Steps 1-5 for each desired limit line.

Displaying and Testing with Limit Lines

After creating limit lines, you can then choose to **display** or **hide** them for each trace. The specified limits remain valid even if limit lines are not displayed.

Limit testing cannot be performed on memory traces.

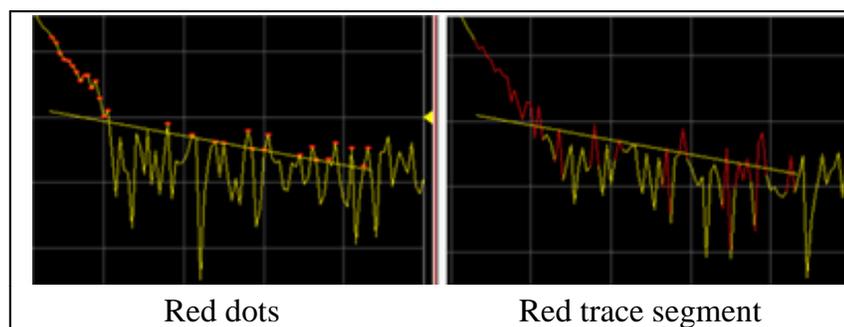
You can choose to provide a visual and / or audible PASS / FAIL indication.

With limit testing turned ON:

- Any portion of the measurement trace that **fails** is **displayed in red**.
- Any portion of the measurement trace that does **NOT fail** remains unchanged and silent.

Display failed trace points or trace segments

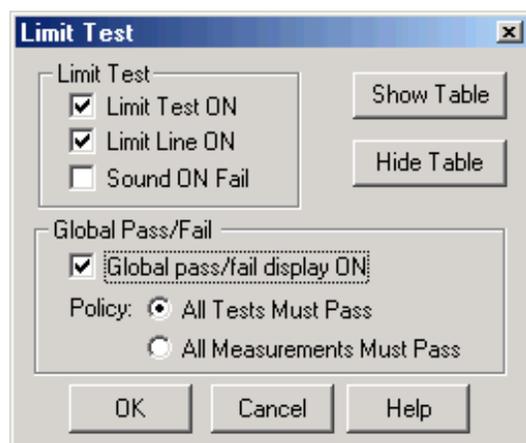
You can display the data points that fail limit line testing as red dots or as a red trace segment. The default behavior can be changed with a Preference setting. [Learn how.](#)



PASS is the default mode of Pass / Fail testing.

A data point will FAIL only if a measured point falls outside of the limits.

- If the limit line is set to OFF, the entire trace will PASS.
- If there is no measured data point at a limit line stimulus setting, that point will PASS.



Limit Test dialog box help

Show Table Shows the table that allows you to create and edit limits.

Hide Table Makes the limits table disappear from the screen.

Note: To ADD a limit line to the table, change the last limit line to either MAX or MIN

Limit Test

Limit Test ON Check the box to compare the data trace to the limits and display PASS or FAIL.

Limit Line ON Check the box to make the limits visible on the screen. (Testing still occurs if the limits are not visible.)

Sound ON Fail Check the box to make the PNA beep when a point on the data trace fails the limit test.

Global Pass/Fail

The Pass/Fail indicator provides an easy way to monitor the status of ALL measurements.

Global pass/fail display ON Check to display the Global Pass/Fail status.

Policy: Choose which of the following must occur for the Global Pass/Fail status to display PASS:

- All Tests (with **Limit Test ON**) Must Pass - This setting reads the results from the Limit Tests. If all tests (with **Limit Test ON**) PASS, then the Global Pass/Fail status will PASS.
- All Measurements Must Pass - This more critical setting shows FAIL unless all measured data points fall within established test limits **and** Limit Test is ON. **Note:** In this mode, if one measurement does NOT have **Limit Test ON**, Global Pass/Fail will show FAIL.

[Learn more about displaying and testing with Limits \(scroll up\)](#)

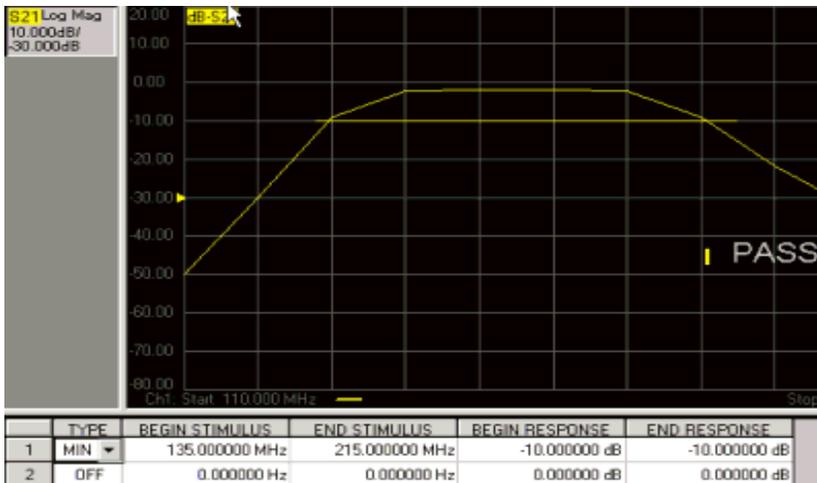
Testing with Sufficient Data Points

Limits are checked only at the actual measured data points. Therefore, It is possible for a device to be out of specification without a limit test failure indication if the data point density is insufficient.

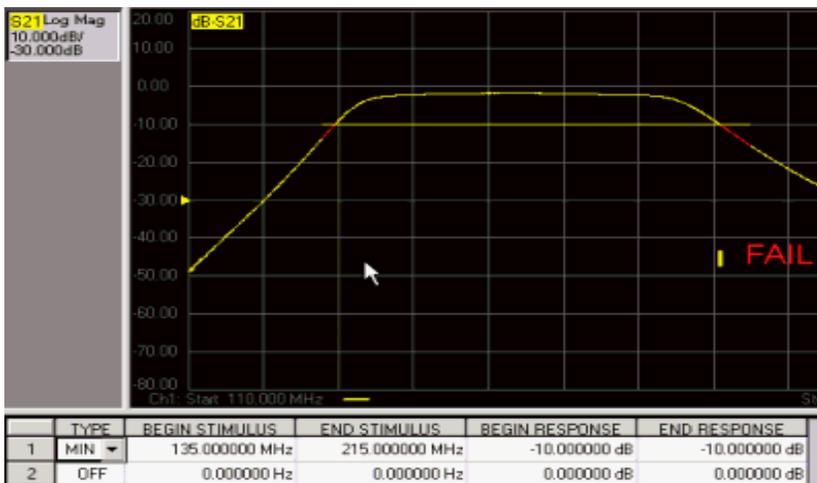
The following image is a data trace of an actual filter using 11 data points (approximately one every vertical graticule). The filter is being tested with a minimum limit line (any data point under the limit line fails).

Although the data trace is clearly below the limit line on both sides of the filter skirts, there is a PASS

indication because there is no data point being measured at these frequencies.



The following image shows the exact same conditions, except the number of data points is increased to 1601. The filter now fails the minimum limit test indicated by the red data trace.



Last Modified:

- 14-Mar-2013 Edited default fail type
- 11-Aug-2009 Added failed dots
- 3-Sep-2008 Removed legacy content
- 2-Feb-2007 MX Added UI

Save and Recall a File

The PNA allows you to save and recall files to and from an internal or external storage device in a variety of file formats.

- [How to Save PNA Instrument State](#)
- [How to Save PNA Measurement Data](#)
- [How to Recall a File](#)
- [About Instrument State and Calibration Data](#) (.csa, .cst, .sta, .cal)
- [About Measurement Data Files](#) (.prn, .snp, .cti, .csv, .mdf)
- [Managing Files without a Mouse](#)

Other Data Outputting topics

How to Save Instrument State and Calibration Files

Use one of the following methods:

Using front-panel HARDKEY [softkey] buttons

1. Press **SAVE**
2. then [**Save**], [**Save As**], or [**Auto Save**]

Using a mouse with PNA Menus

1. Click **File**
2. then **Save** or **Save As**

[Learn all about PNA Instrument State files.](#)

Save Immediately saves the PNA state and possibly calibration data to the filename and extension you used when you last performed a Save. This file will be overwritten the next time you click **Save**. To prevent this, use one of the following methods.

Save As Starts the [Save As](#) dialog box.

Auto Save NOT available from the PNA menu. Saves state and calibration data to the internal hard disk in the c:\users\public\network analyzer\documents folder. A filename is generated automatically using the syntax "**atxxx.csa**"; where xxx is a number that is incremented by one when a new file is Auto Saved.

Save (State and Calibration) As dialog box help



Save Allows you to navigate to the directory where you want to save the file.

File name Displays the filename that you either typed in or clicked on in the directory contents box.

Note: Filenames (not including the path name) MUST be limited to 64 characters.

Save as type

The following file types save **Instrument states and Calibration data**. You can save, and later recall, instrument settings and calibration data for **all channels** currently in use on the PNA. These file types are only recognized by Agilent PNA Series analyzers.

[Learn more about these file types.](#)

- ***.csa** - save Instrument state and actual Cal Set data (cal/state archive) **Default selection.**
- ***.cst** - save Instrument state and a link to the Cal Set data.
- ***.sta** - save Instrument state ONLY (**no** calibration data)
- ***.cal** - save actual Calibration data ONLY (**no** Instrument state)

Note: To save the PNA screen as .bmp, .jpg, or .png graphics file types, click **File / Print to File.**

[Learn more.](#)

Save Saves the file to the specified file name and directory.

How to Save Measurement Data

Use one of the following methods:

Using front-panel HARDKEY [softkey] buttons

1. Press **SAVE**
2. then **[Save Data As]**

Using a mouse with PNA Menus

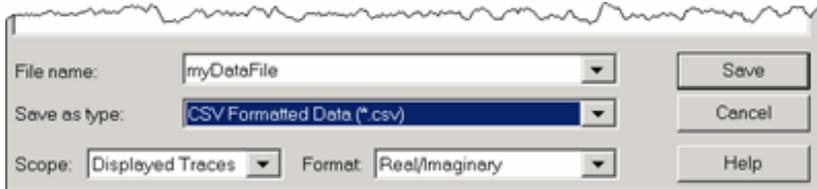
1. Click **File**
2. then **Save Data As**

Save Data As Saves the current trace(s) to the specified type of file.

Note: This dialog now contains the settings previously selected from the old [Define Data Save](#) dialog.

◀ Programming Commands ▶

Save Data As dialog box help



Note: Before saving measurement data, always [trigger a single](#) measurement, and then allow the PNA channel to go into Hold. This ensures that the entire measurement trace is saved.

Note: [Memory traces](#) can NOT be saved to any file type (PRN, SNP, CTI, CSV, MDF).

Save Allows you to navigate to the directory where you want to save the file.

File name Displays the filename that you either typed in or clicked on in the directory contents box.

Note: Filenames (not including the path name) MUST be limited to 64 characters.

Save as type Choose from: (click each to learn more about each file type): [*.prm](#), [*.SNP](#), [*.SNPX](#), [*.cti \(citifile\)](#), [*.csv](#), [*.mdf](#).

- FCA, GCA, Swept IMD, and Swept IMDx data can be saved to a special csv format. Learn how ([FCA](#), [GCA - Swept IMD](#))
- **Trace and Noise Parameters (*.snp)** - Save the noise figure parameters and S-parameters. [Learn more.](#)

- To save the PNA screen as .bmp, .jpg, or .png graphics file types, click **File / Print to File**. [Learn more.](#)

Data Scope

Determines what traces are saved to a file. Available ONLY with *.cti, *.csv, and *.mdf.

- **Auto**
 - When correction is OFF, saves the specified trace.
 - When correction is ON, saves all corrected parameters associated with the calibrated ports in the Cal Set.
 - For GCA and Swept IMD channels, saves the active trace only.
- **Single Trace** - Saves the active trace.
- **Displayed Traces** - Saves all displayed traces for all channels.

Format

Determines the format of the data. Available with (CTI Formatted, CSV, SNP, MDIF)

- **Auto** - Data is saved in LogMag or LinMag if one of these is the currently selected display format. If format is other than these, then data is saved in Real/Imag.
- **Displayed Format (CSV and MDIF only)** - Data is saved in the format of the displayed trace.
- **LogMag, LinMag, Real/Imag** - Select output format.
 - The imaginary portion for all **LogMag** and **LinMag** data is saved in degrees.
 - Real/ Imag data is never smoothed.

Note: .prn files can only save the active trace in the displayed format.

Save Saves the file to the specified file name and directory.

How to Recall (open) a file

Use one of the following methods:

Using front-panel HARDKEY [softkey] buttons

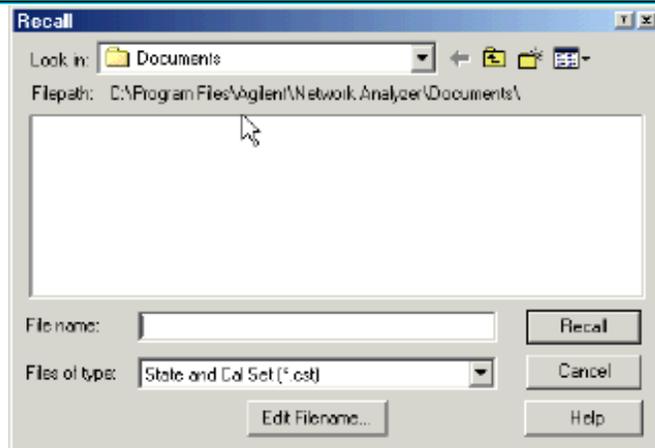
1. Press **RECALL**
2. then [**Recall**]
3. Select from a list of 12 files shown on softkeys. The list can be sorted by 'most recently used' or alphabetically depending on a preference. The preference setting appears at the bottom of the second page of softkeys listing files to be recalled, or on the [Preference dialog](#).
4. or press the Recall softkey to start the following dialog.

Using a mouse with PNA Menus

1. Click **File**
2. then **Recall**
3. Select from the top 6 'most recently used' list.
4. or click **Recall** to start the following dialog.

Programming Commands

Recall dialog box help



Look in Allows you to select the directory that contains the file that you want to recall.

Filename Displays the filename that you either typed in or clicked on in the directory contents box.

Files of type Allows you view and select files that are listed in categories of a file type. The following types of files can be recalled into the PNA: All [State files](#), Citi files, SNP files.

Recalling PNA State files

When an Instrument State file is recalled into the PNA, the current state of the instrument is overwritten with the recalled state. A *.cal file does not contain an instrument state, but only calibration data. [Learn more about Instrument States.](#)

See also [Power ON and OFF during Save / Recall, User Preset, and Preset.](#)

Recalling Data files into the PNA

Citi files and SNP files can be recalled and viewed in the PNA.

1. Click **File** then **Recall**.
2. Select **Citifile Data** or **Snp**.
3. Select the file to recall
4. Click **Recall**.

Note: Citi files that were saved in [CW Time sweep](#) can NOT be recalled into the PNA

Note: Filenames (not including the path name) that are longer than 64 characters will NOT be recalled.

Recalled data is ALWAYS displayed on the PNA using [LogMag format](#), regardless of how the file was stored.

The channel is placed in Trigger Hold. If triggering is resumed, the data will be overwritten.

SNP files are recalled as traces into a single window and channel, beginning at the [highest available channel number allowed on the PNA](#). For multi-port SNP files (greater than 4 ports), if the number of S parameters in the file is beyond the [maximum number of traces in a window](#), then new windows will be created.

Citi files are recalled into the same window and channel configuration as when they were saved. However, the new recalled channel numbers begin with the [highest channel number allowed on the PNA](#) and decrement for each additional channel.

For example, when a citi file is saved, two traces are in window 1, channel 1 and two additional traces are in window 2, channel 2. When recalled into a factory preset condition (1 trace in window 1, channel 1), the first two recalled traces appear in window 2, highest channel number, and the second two traces appear in window 3, (highest channel number -1). See also [Traces, Channels, and Windows on the PNA](#)

Recall Recalls the file displayed in the file name box.

Instrument State / Calibration Files

You can save, and later recall, instrument settings and calibration data **for all channels** currently in use on the PNA.

An **Instrument State** contains almost every PNA setting. The following PNA settings are NOT saved and recalled with Instrument State:

- [GPIB address](#)
- [RF power ON/OFF](#) (depends on current setting)
- [Test set I/O settings](#)

The following file types are used to save and recall instrument states and Cal Set information:

File Types		Information that is stored for each channel
.csa	.cst	<p>Instrument State Information</p> <p>Channels/Traces Averaging</p> <p>Windows Markers</p> <p>Triggering Math/memory</p> <p>Format Limits</p> <p>Scale More...</p> <p>Stimulus Information:</p> <p>Frequency range Alternate sweep</p> <p>Number of points Port powers</p> <p>IF bandwidth Source attenuators</p> <p>Sweep type Receiver attenuators</p> <p>Sweep mode Test Set port map</p>
		<p>Cal Set Information</p> <p>GUID (Globally Unique Identifier) provides link to Cal Set</p> <p>Name, Description, Modify date</p> <p>Stimulus Information:</p> <p>Frequency range Alternate sweep</p> <p>Number of points Port powers</p> <p>IF bandwidth Source attenuators</p> <p>Sweep type Receiver attenuators</p> <p>Sweep mode Test Set port map</p> <p>Error Terms: Directivity, Crosstalk, Source match, Load match, Reflection tracking, Transmission tracking</p>
	.sta	
		.cal

File Type Descriptions and Recall

The following describes each file type, and what occurs when the file type is recalled.

*.sta files

- Contain ONLY instrument state information - NOT Cal data.
- When recalled, they always replace the current instrument state immediately.

*.cst files

- Contain BOTH instrument state and a LINK to the Cal Sets. [Learn more about Cal Sets.](#)
- The **quickest and most flexible** method of saving and recalling a calibrated instrument state.
- Channels need not have cal data to save as .cst file.
- When recalled, the state information is loaded first. Then the PNA tries to [apply a Cal Set](#) as you would do manually. If the stimulus settings are different between the instrument state and the linked Cal Set, the usual choice is presented ([see Cal Sets](#)). If the linked Cal Set has been deleted, a message is displayed, but the state information remains in place.
- Because only a link to the Cal Set is saved, the Cal Set can be shared with other measurements.

Note: Before saving a .cst file, be sure that a User Cal Set (NOT a Cal Register) is being used for the calibration. Cal Registers are overwritten with new data whenever a calibration is performed, and may not be accurate cal data when the .cst file is recalled. [Learn more about Cal Sets.](#)

*.cal files

- Contain ONLY Cal Set information.
- When recalled, the Cal Set is NOT automatically applied. Apply the calibration data to a channel as you would [apply any Cal Set](#).
- [Learn about Recalling](#)

*.csa files

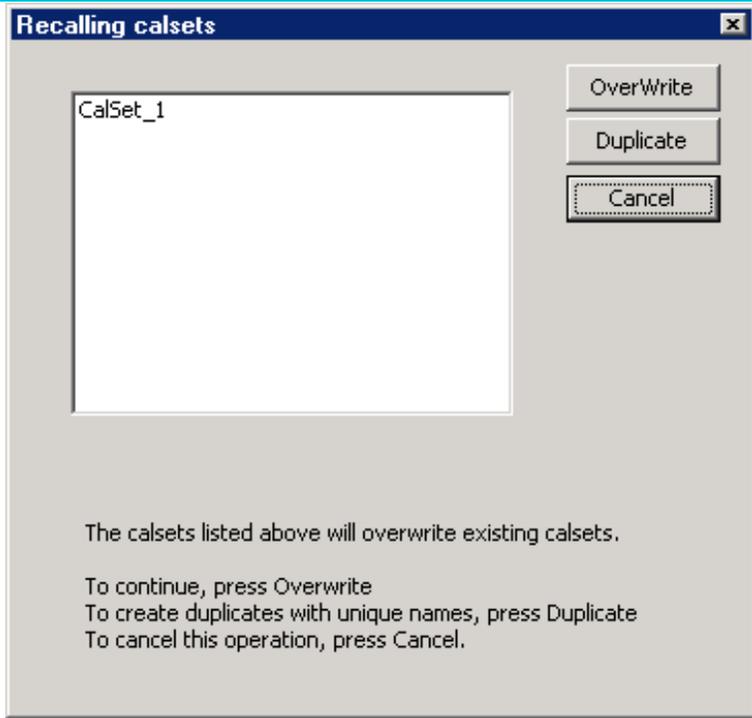
- Contain ALL instrument state and the actual Cal Set; not a link to the Cal Set.

- The **safest** method of saving and recalling a calibrated instrument state. However, the file size is larger than a *.cst file, and the save and recall times are longer. In addition, because the actual Cal Set is saved, it is very difficult to share the cal data with other measurements.
- Channels need not be calibrated to save as .cst file.
- The Cal Set that is saved could be a [Cal Register or a User Cal Set](#).
- [Learn about Recalling](#)

Note: *.pcs files are the internal file format the PNA uses for storing cal sets. These files should never be accessed or copied by the user.

Recalling Cal Sets

Recalling Cal Sets dialog box help



The calsets listed above will overwrite existing calsets.

To continue, press Overwrite
 To create duplicates with unique names, press Duplicate
 To cancel this operation, press Cancel.

Both **.cal** and **.csa** file types contain whole Cal Sets. When these file types are recalled, the PNA checks to see if the incoming Cal Set GUID matches an existing PNA Cal Set GUID. If it does, and if the rest of the Cal Set contents are different in any way, then both of these Cal Sets can NOT coexist in the PNA and you are offered the following choices.

Because all PNA channels are saved, there could be more than one Cal Set in either of these file types.

Overwrite The incoming Cal Set will replace the existing Cal Set.

Duplicate (Only available with .cal recalls.) Because the Cal Set is not automatically applied, you can choose to apply either the original or duplicate Cal Set. The original Cal Set remains in the .cal file.

Cancel Abandon the recall operation.

The PNA will offer a choice as described in each file type below. [Learn more about Cal Sets.](#)

Measurement Data Files

Measurement data is saved as ASCII file types for use in a spreadsheet or CAE programs.

Note: Before saving measurement data, always [trigger a single](#) measurement, and then allow the PNA channel to go into Hold. This ensures that the entire measurement trace is saved.

Note: [Memory traces](#) can NOT be saved to any file type (PRN, SNP, CTI, CSV, MDF).

The following file types can be saved by the PNA.

- [*.prn files](#)
- [*.SNP \(Touchstone\)](#)
- [*.cti \(Citifile\)](#)
- [*.csv](#)
- [*.mdf \(MDIF\)](#)

*.prn Files

Prn files have the following attributes:

- Comma-separated data which can be read into rows and columns by spreadsheet software, such as Microsoft Excel. To avoid the "delimiting" dialog boxes, change the filename extension from .prn to .csv. Then open directly into Microsoft Excel.
- Contain formatted and corrected stimulus and response data for the current active trace ONLY.
- Are Output only - they cannot be read by the PNA.
- Beginning with PNA Rev 6.2, [FCA](#) and [Cal Set Viewer](#) data can be saved to *.prn files

Example:

"S11 Log Mag"

"Frequency (Hz)",	"dB"
3.000000e+005 ,	-3.528682e+001 ,
4.529850e+007 ,	-2.817913e+001 ,
9.029700e+007 ,	-3.216808e+001 ,
1.352955e+008 ,	-3.101017e+001 ,

.SNP Format (*.s1p, *.s2p, *.s3p, *.s4p, and so forth)

- *.SNP file format, also known as Touchstone format, is specified by IBIS. [See the Touchstone specification.](#)
- *.SNP file format is used by CAE programs such as Agilent's Microwave Design System (MDS) and Advanced Design System (ADS).
- *.SNP data is saved using the **File**, [Save Data As](#) dialog.

Before saving measurement data, always [trigger a single](#) measurement, and then allow the PNA channel to go into Hold. This ensures that the entire measurement trace is saved.

*.SNP files and other PNA settings

- .SNP data can be [recalled](#) and viewed on the PNA, or read by the PNA [embed/de-embed](#) functions.
- To save SNP data with an [external test set](#) enabled, at the File, [Save As](#) dialog, select **SNP File(*.s*p)**, then complete the ["Choose Ports " dialog](#).
- When [Fixturing](#) is enabled, all of the enabled data transforms (De-embedding, Port Z Conversion, and so forth) are applied to saved SNP files.
- When [Smoothing](#) is applied to a trace, the smoothing is NOT saved when the format is Real, Imaginary (RI). Select a different format to save the smoothed data.
- Segmented FCA data is saved to *.S2PX files. Scroll down or [click here](#) to learn more.
- Learn about [FCA parameters that are saved to an S2P file](#).
- Noise Figure parameters and S-parameters can be saved to an *.s2p file. [Learn more](#).
- Balanced parameters can be saved to *.SNP files. See the ["Choose Ports " dialog](#).
- **IMPORTANT** - ALL valid data is saved using the same format and settings (trace math, offset, delay, and so forth) as the active measurement. This can cause the data that is saved for the non-active measurements to be dramatically different from the data that is displayed. For example, when saving an S2P file, if the active S11 measurement is set to Data/Mem (data divided by memory), then ALL 4 S-parameters are saved using Data/Mem. The memory trace that is used in the Data/Mem operation is the same as that used in the active (S11) measurement.

What is Saved

*.SNP data is generally used to gather all S-parameters for a fully corrected measurement. The PNA saves the data that is available on the channel of the active measurement.

File Type	# of Ports	# of S-parameters saved
*.s1p	1	1 S-parameter
*.s2p	2	4 S-parameters
*.s3p	3	9 S-parameters
*.s4p	4	16 S-parameters
...
*.SNP	N	N^2 S-parameters

- If correction for a **Full N-port cal** is applied, then valid data is returned for all corrected s-parameters. Response calcs will save uncorrected data.
- If requesting **less** data than is available, the [Choose ports for SNP data](#) dialog appears. Previous to PNA release 6.2, data was returned beginning with the first calibrated ports until your request is fulfilled.
- If correction is NOT applied, the PNA returns as much applicable raw data as possible using S-parameter measurements on the selected channel. Data that is not available is zero-filled. For example, if correction is NOT applied and the active measurement is S11, and an S21 measurement also exists on the channel, then data is returned for the S11 and S21 measurements. Data for S12 and S22 is not available and therefore returned as zeros in Real/Imaginary format. In Log Mag/Phase format, this appears as -200 dB and 45 degrees.

.SNP Data Output

.SNP files contain header information, stimulus data, a response data pair for EACH S-parameter measurement. The only difference between .s1p, s2p, and so forth, is the number of S-parameters that are saved.

The following is a sample of **Header information**:

```
!Agilent Technologies,E8362B,US42340026,Q.03.54
!Agilent E8362B: Q.03.54
!Date: Friday, April 25, 2003 13:46:41
!Correction: S11(Full 2 Port SOLT,1,2) S21(Full 2 Port SOLT,1,2) S12(Full 2 Port
SOLT,1,2) S22(Full 2 Port SOLT,1,2)
!S2P File: Measurements:S11,S21,S12,S22:
# Hz S R I R 50
```

Note: Although the following shows Real / Imag pairs, the format could also be LogMag / Phase or LinMag / Phase

***.s1p Files**

Each record contains 1 stimulus value and 1 S-parameter (total of 3 values)

Stim Real (Sxx) Imag(Sxx)

***.s2p Files**

Each record contains 1 stimulus value and 4 S-parameters (total of 9 values)

Stim Real (S11) Imag(S11) Real(S21) Imag(S21) Real(S12) Imag(S12) Real(S22) Imag(S22)

***.s3p Files**

Each record contains 1 stimulus value and 9 S-parameters (total of 19 values)

Stim Real (S11) Imag(S11) Real(S12) Imag(S12) Real(S13) Imag(S13)
Real (S21) Imag(S21) Real(S22) Imag(S22) Real(S23) Imag(S23)
Real (S31) Imag(S31) Real(S32) Imag(S32) Real(S33) Imag(S33)

***.s4p Files (and so forth...)**

Each record contains 1 stimulus value and 16 S-parameters (total of 33 values)

Stim Real (S11) Imag(S11) Real(S12) Imag(S12) Real(S13) Imag(S13) Real(S14) Imag(S14)
Real (S21) Imag(S21) Real(S22) Imag(S22) Real(S23) Imag(S23) Real(S24) Imag(S24)
Real (S31) Imag(S31) Real(S32) Imag(S32) Real(S33) Imag(S33) Real(S34) Imag(S34)
Real (S41) Imag(S41) Real(S42) Imag(S42) Real(S43) Imag(S43) Real(S44) Imag(S44)

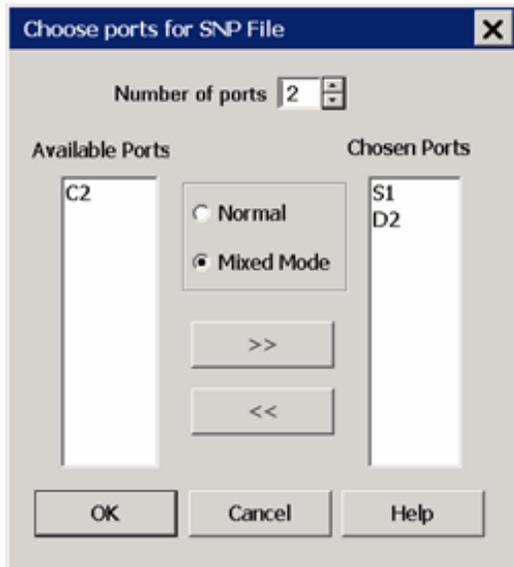
.S2PX Data Output

*.S2PX files are used for Segmented Mixer Data. [Learn more.](#)

The following ADDITIONAL columns precede parameter data:

SegIndex,InputFreq,OutputFreq,LO1Freq,InputPower,LO1Power, <parameter data>

Choose ports for SNP File dialog box help



This dialog allows you to choose which S-parameter data to save when selecting File, [Save As](#), Trace SNP and any of the following conditions exist:

- you request less data than is available
- you want data for more than 4 ports
- a balanced measurement is active

Number of ports Select the number of ports for which data will be saved.

The following buttons appear when a [Balanced measurement](#) is displayed.

Normal Click to save normal (single-ended) port data.

Mixed Mode Click to save balanced (logical) port data. Choices are based on the [topology selection](#) for current active parameter:

- **SE-Bal**: Choose from S1, D2, C2 (Single-ended port 1, Differential port 2, Common port 2)
- **SE, SE, Bal**: Choose from S1, S2, D3, C3 (Single-ended port 1, Single-ended port 2, Differential port 3, Common port 3)
- **Bal-Bal**: Choose from D1, C1, D2, C2 (Differential port 1, Common port 1, Differential port 2, Common port 2)

For example, with SE-Bal topology, choose 2 ports, S1 for first, and D2 for second. The following 4 parameters are saved: Sss11, Ssd12, Sds21, Sdd22.

Arrow buttons Click to Add and Remove ports for the following columns:

Available Ports The PNA / External test set ports. There may NOT be valid data available for all of these ports. [Learn more.](#)

Chosen Ports When **OK** is clicked, SNP data is saved for these ports.

OK Becomes available when the number of **Chosen ports** = the **Number of ports** to save. Click to save to SNP file.

With **Number of ports** = 2, .s2p data is saved; with **Number of ports** = 3, .s3p data is saved, and so forth. [Learn more about SNP files](#)

.cti CitiFiles

Citifile format is compatible with the Agilent 8510 Network Analyzer and Agilent's Microwave Design System (MDS).

You can do the following using citifiles :

- save the active trace, or all traces.
- save formatted or unformatted citifile data

To save Formatted *.cti data

1. Click **File** then **Save Data As**
 2. Select **Citifile Formatted Data (*.cti)**
-

```
CITIFILE A.01.01
!Agilent Technologies,E8364B: A.07.50.
!Format: LogMag/Angle
!Date: Thursday, April 17 2014
NAME CH1_DATA
VAR Freq MAG 3
DATA S11 DBANGLE
VAR_LIST_BEGIN
10000000
25005000000
50000000000
VAR_LIST_END
BEGIN
-14.837049,-54.52504
12.872375,105.69022
12.423248,-142.31508
END
CITIFILE A.01.01
!Agilent Technologies,E8364B: A.07.50.
!Format: LinMag/Angle
!Date: Thursday, April 17 2014
NAME CH2_DATA
VAR Freq MAG 3
DATA SC21 MAGANGLE
VAR_LIST_BEGIN
10000000
25005000000
50000000000
VAR_LIST_END
BEGIN
0.00012980904,85.130768
0.0006259337,-56.60146
0.0062053087,-99.910851
END
```

The above image is a Citifile opened in Notepad. There are two traces in separate channels - one is an FCA trace. Each trace has 3 data points. The save settings = **Displayed Traces Content**, and **Auto Format**.

Format is identified by DBANGLE (log mag), MAGANGLE (Lin Mag), or RI (real, imaginary - NOT shown)

On the [data access map](#), Formatted data is taken from location 2 or 4.

To save Unformatted *.cti data

1. Click **File**, then **Save Data As**
2. Select **Citifile Data Data (*.cti)**

On the [data access map](#), Unformatted data is taken from the block just before Format.

Citifiles can be recalled and viewed in the PNA. [Learn more.](#)

*.csv Files

Note: 2D Gain Compression data is saved as *.csv files using a different format than shown here. [Learn more.](#)

CSV files are read by spreadsheet programs such as Microsoft Excel.

To save *.csv files:

1. Click **File**, then **Save Data As**
2. Under **Save as type**, select **CSV Formatted Data**

*.csv files contain: header information and the following Comma-Separated Values.

- Stimulus data
- Data pairs for EACH S-parameter

```
!CSV A.01.01
!Agilent Technologies, N5242A, US473000004, A.08.20.01
!Format: LogMag/Angle
!Date: Thursday, October 23, 2008 11:14:01

BEGIN CH1_DATA
!Freq (Hz), S11 (DB), S11(DEG), S12(DB), S12(DEG), S21 (DB), S21(DEG), S22(DB),
S22(DEG)
1e+009, 0.042540751, -0.024109257, 0.56931657, 0.042540751, -
0.024109257, 0.56931657_
...
...
...
END

BEGIN CH2_DATA
!Freq (Hz), S11 (DB), S11(DEG), S12(DB), S12(DEG), S21 (DB), S21(DEG), S22(DB),
S22(DEG)
1e+009, 0.042540751, -0.024109257, 0.56931657, 0.042540751, -
0.024109257, 0.56931657_
...
...
...
END
```

*.mdf Files

MDIF files are compatible with Agilent ADS (Advanced Design System). [Learn more at the Agilent website.](#)

To save *.mdf files:

1. Click **File**, then **Save Data As**

2. Under **Save as type**, select **MDIF Data**

*.mdf files contain: header information and space-separated data:

- Stimulus data
- Real and Imaginary data pair for EACH S-parameter measurement

```
!MDF A.01.01
!Agilent Technologies, N5242A, US473000004, A.08.20.01
!Format: LogMag/Angle
!Date: Thursday, October 23, 2008 11:14:01

BEGIN CH1_DATA
%Freq (real) S[1,1] (complex) S[1,2] (complex) S[2,1] (complex) S[2,2] (complex)
1e+009 0.042540751 -0.024109257 0.56931657 0.042540751, -
0.024109257 0.56931657
...
...
...
END

BEGIN CH2_DATA
%Freq (Hz) S[1,1] (complex) S[1,2] (complex) S[2,1] (complex) S[2,2] (complex)
1e+009 0.042540751 -0.024109257 0.56931657 0.042540751 -
0.024109257 0.56931657
...
...
...
END
```

Define Data Saves

Note: Although these settings are still supported, they are no longer necessary to save data files. The [Save Data As](#) dialog box contains these settings.

How to select Define Data Saves

Using front-panel HARDKEY [softkey] buttons	Using a mouse with PNA Menus
1. Press SYSTEM	1. Click Utility
2. then [Configure]	2. then System
3. then [More]	3. then Configure
4. then [Preferences]	4. then Preferences
5. then Data Saves	5. then Data Saves

Programming Commands

Define Data Saves dialog box help

Note: Although these settings are still supported, they are no longer necessary to save data files. The [Save Data As](#) dialog box contains these settings.

The following settings survive an Instrument Preset and PNA Shutdown.

CitiFile, CSV, and MDIF Contents

Determines what is saved to a .cti file.

Auto - Saves the active trace. Additional traces are saved if correction is ON. For Full 2-port calibration, 4 traces are saved; for Full 3-port calibration, 9 traces are saved, and so forth.

Single Trace - Saves the active trace.

Displayed Traces - Saves all displayed traces for all channels.

Citifile and CSV Format

Auto - Data is saved in LogMag or LinMag if one of these is the currently selected display format. If format is other than these, then data is saved in Real/Imag.

LogMag, LinMag, Real/Imag - Select output format.

- The imaginary portion for all LogMag and LinMag data is saved in degrees.
- Real/ Imag data is never smoothed.

SnP Format (.s1p, .s2p, .s3p)

[Learn more about SnP files.](#)

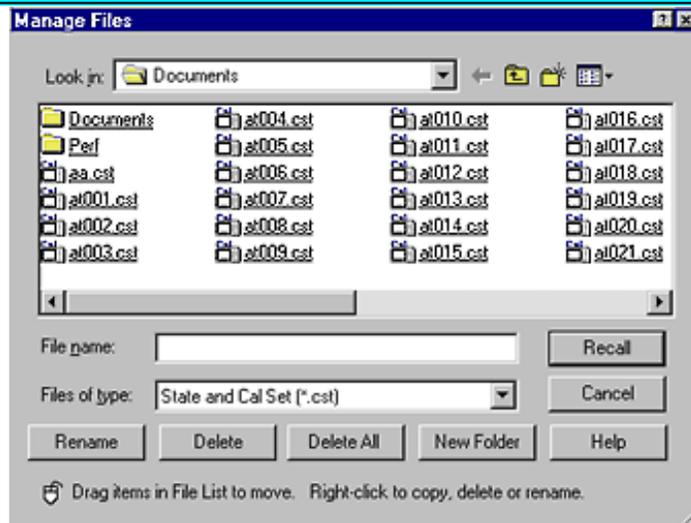
Auto - Data is saved in LogMag or LinMag if one of these is the currently selected format. If format is other than these, then data is saved in Real/Imag.

LogMag, LinMag, Real/Imag - Select output format. The imaginary portion for all LogMag and LinMag data is output is in degrees.

Manage Files without a Mouse

The Manage Files dialog box is designed to be used from the front panel. It performs the same function as Windows Explorer, but can be used without the use of a mouse or keyboard.

Manage Files dialog box help



Recall Opens a Network Analyzer file already stored in memory.

Rename Renames a file that is selected in the open folder.

Delete Removes a selected file from the open folder.

Delete All Removes all files of the file type selected that appear in the open folder.

New folder Create a new folder and give it a name.

◀ **Programming Commands** ▶

Last modified:

25-Nov-2013 Updated for Win7 folder
3-Apr-2013 Removed trademarks
30-Jan-2013 Added link to Touchstone IBIS spec
1-Nov-2012 Added Trace and Noise (snp)
29-Jun-2012 Added sNp data with response cal
29-Jun-2012 Added filename limitation
1-Jun-2011 Added S2PX
20-Sep-2010 Added memory trace notes
11-Nov-2009 Added snp fixturing note
3-Sep-2008 Removed legacy content
22-Aug-2008 Clarified invalid SNP data as -200 dB.
9-Jun-2008 Added no smoothing for real/imag cti files.
12-May-2008 Removed csv detail
17-Apr-2008 Added clarification and image to define data saves.
17-Oct-2007 Added note for MM
10/23/06 Added pcs note
9/18/06 MQ Added choose ports for snp
9/12/06 Added link to programming commands

Drive Mapping

Drive mapping allows you to share disk drives between the PNA and an external computer.

To prepare for Drive Mapping:

1. Both the PC and PNA must be connected to a shared computer network
2. You must know the full computer name of the Analyzer you are mapping **TO**. [Tell me how](#)

Note: This procedure requires a mouse and keyboard.

Map to a drive on the PNA from an External PC

1. On the analyzer desktop, click **Windows Explorer**. Right-click on the drive you want to share. Click **Sharing...**
2. In the dialog box, select **Shared this folder**. In the **Share Name** box, type in a share name for the drive. For example: **C\$**. Click **OK**.
3. On the external PC desktop, click **Windows Explorer**. From the **Tools** menu, click **Map Network Drive**.
4. If the current logon on your PC is different from the current logon on the analyzer, click **Connect using a different Logon** to connect to using the current analyzer logon, .This logon must be registered on the external PC. To see the current logon on either the PC or analyzer, hold **Ctrl - Alt**, and press **Delete**.
 - a. In the **Connect as** box, type the logon currently being used by the analyzer.
 - b. In the **Password** box, type the logon password that you use on the external computer. Click **OK**
5. In the **Folder** box, type *//computername ([prep1](#))/share name* (from step 2). (For example: *//SLT1234/C\$*)
6. Click **Finish**.

Last Modified:

25-Nov-2013 Modified for Win7

Print a Displayed Measurement

The analyzer allows you to print a displayed measurement to a printer or to a file. The printer can be either networked or local.

- [Connecting a Printer](#)
- [Printing](#)

Other Outputting Data topics

Connecting a Printer

You can connect a printer to one of the PNA USB ports or to the LAN connector.

To Add a Printer

Note: If you try to print from the PNA application and the **Add Printer Wizard** appears, click **Cancel** and add the printer using the following procedure.

1. From the PNA application, click **View** then click **Minimize Application**
2. On the Windows taskbar, click **Start**, point to **Settings**, then click **Printers**.
3. Double-click **Add Printer**.
4. Follow the instructions in the **Add Printer** Wizard.

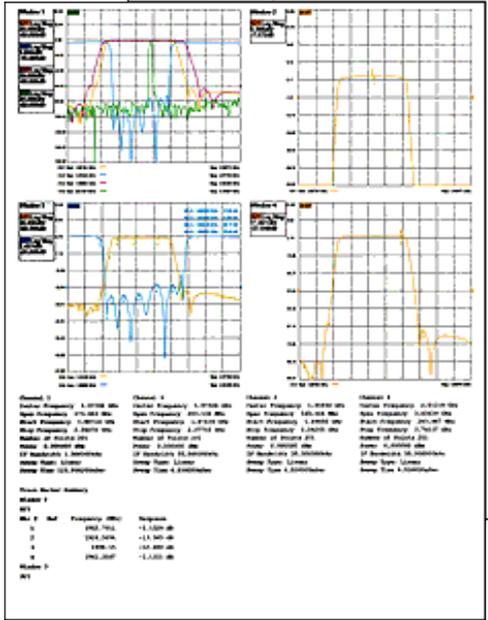
For more information, refer to Microsoft Windows Help or your printer documentation.

Printing

- [Print a Hardcopy](#)
- [Page Setup](#)
- [Print to File](#)

The measurement information on the screen can be printed to any local or networked printer that is connected to the PNA. The graphic below shows an example of how a screen-capture image appears when printed. The [Page Setup](#) settings allows you to customize the printed form of the measurement information.

Ch	Ref	Frequency (MHz)	Response
1	100%	1.000000	1.76e-06
2	100%	1.000000	1.84e-06
3	100%	1.000000	1.84e-06
4	100%	1.000000	1.84e-06
5	100%	1.000000	1.84e-06
6	100%	1.000000	1.84e-06
7	100%	1.000000	1.84e-06
8	100%	1.000000	1.84e-06
9	100%	1.000000	1.84e-06
10	100%	1.000000	1.84e-06



How to Print a Hardcopy

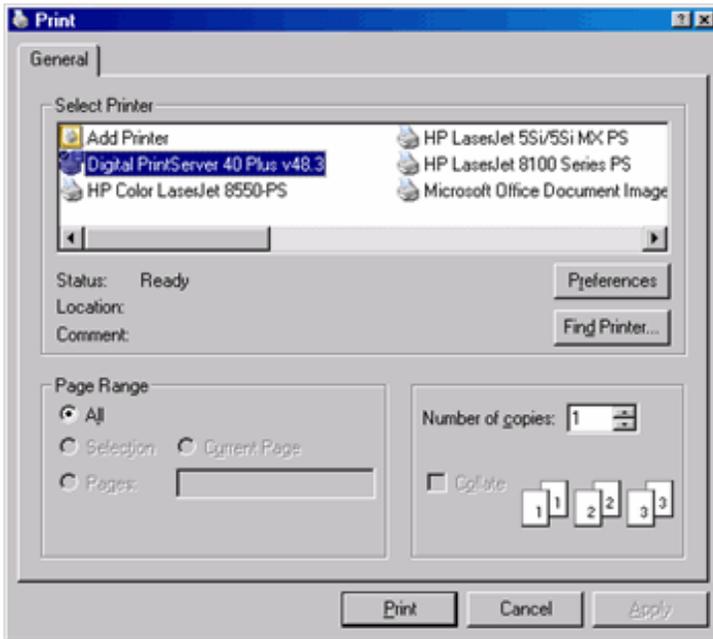
Using front-panel HARDKEY [softkey] buttons

1. Press **PRINT**
2. then **[Print]**

Using a mouse with PNA Menus

1. Click **File**
2. then **Print**
3. then **Print**

No programming commands are available for this feature.



Note: For information on the choices in the Print dialog box, see Windows Help.

Page Setup

The Page Setup dialog allows flexibility in the appearance that measurement data is printed. After setting up the page, click **File**, then **Print...** to obtain a hard-copy.

How to select Page Setup

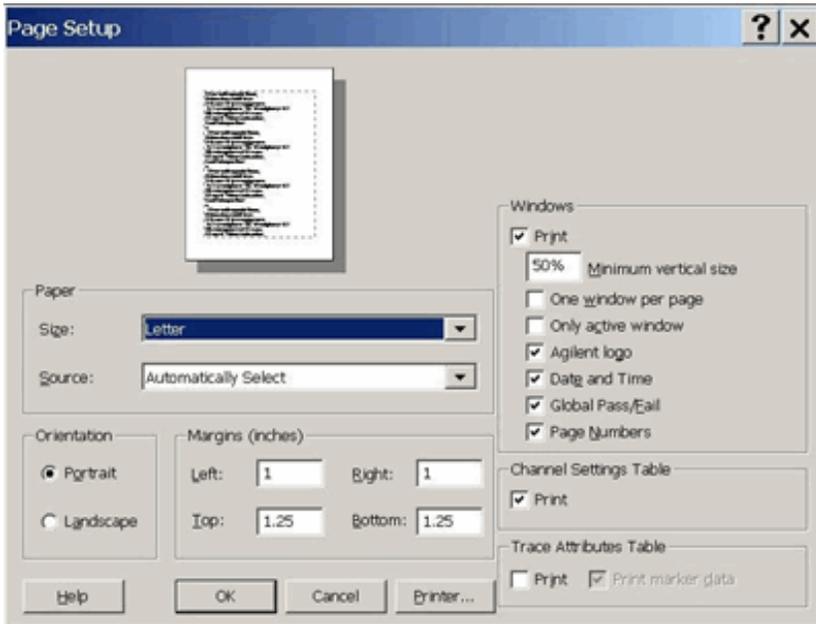
Using front-panel HARDKEY [softkey] buttons

1. Press **PRINT**
2. then **[Page Setup]**

Using a mouse with PNA Menus

1. Click **File**
2. then **Print**
3. then **Print Options**

◀ **Programming Commands** ▶



Page Setup dialog box help

Paper, Orientation, and Margins

These settings do NOT survive a PNA shutdown.

See Windows Help for information on these settings.

Windows

The following PNA-specific settings DO survive a PNA shutdown:

Minimum vertical size Adjust to change the amount of a page that the measurement window fills. The adjustment range is from 40 to 100%.

One window per page Check to print one window per page. Clear to print all selected windows without a forced page break.

Only active window Check to print only the active window. Clear to print all windows.

Agilent logo Check to print the Agilent logo to the header.

Data and Time Check to add the current date and time to the header.

Global Pass/Fail Check to add the Global Pass/Fail status to the header.

Page Numbers Check to add page numbers (1 of n) to the header.

Channel Settings Table

Print Check to print the channel settings table.

Segment data can no longer be printed.

Trace Attributes Table

Print Check to print the Trace Attributes Table. The Trace Attributes are measurement type, correction factors ON or OFF, smoothing, options, and marker details. The Trace Attributes are listed by Trace ID# for each window.

Each Trace ID# can have multiple entries depending on the number of markers associated with the trace. The marker details are marker number, position and response. If there are multiple markers on a trace, the trace attributes are only shown for the first marker. However, the trace attributes for the first marker apply to all other markers on that trace.

The options column can have one or more options. **D** for Delay, **M** for Marker, **G** for Gating. Multiple options selected would appear as follows: DMG.

Print marker data Check to print all marker data. The amount of data depends on how many markers are created.

Print to a File

The analyzer can save a screen-capture image in any of the following formats:

- **.png** (preferred format)
- **.bmp** (bitmap)
- **.jpg**

The analyzer automatically saves the file to the current path. If not previously defined, the analyzer automatically selects the default path c:\users\public\network analyzer\documents.

A .bmp file, like a .prn file, can be imported into software applications such as Microsoft Excel, Word, or Paint to display a screen-capture image.

[See Save and Recall files for more information.](#)

How to Print to a File

Using front-panel HARDKEY [softkey] buttons

1. Press **PRINT**
2. then **[Print to File]**

Using a mouse with PNA Menus

1. Click **File**
2. then **Print**
3. then **Print to File**

Programming Commands

Last modified:

- | | |
|-------------|--------------------------------|
| 25-Nov-2013 | Default folder for Win7 |
| 13-Apr-2011 | Removed segment data checkbox |
| 3-Sep-2008 | Removed legacy content |
| 23-Apr-2008 | Added preference survive note. |
| 10/19/06 | Modified for new print dialog |

Programming Guide

Two ways to find programming commands:

1. From a simulated PNA User Interface:

File Trace/Chan Response Cal Marker/Analy Stimulus Utility

	GPIB / SCPI	COM
2.	Command Tree	COM Object Model
See Also	Example Programs Learning about GPIB	Example Programs Learning about COM

See Also

- [New Programming Commands](#)
- [Remotely Specifying a Source Port](#)
- [Shut Down or Restart the PNA Remotely](#)
- [Your Programs on Windows 7](#)
- [LXI and VXI-11.3 Compliance](#)
- [VEE Examples](#) with runtime installed.
- [Using Macros](#)
- [Code Translator App.](#)
- [Superseded / Replacement Commands](#)
- [Data Access Map](#)
- See more PNA programming information and examples at: <http://na.tm.agilent.com/pna/programming/>

PNA Object Model



NaWindow

Traces

Trace

CalKit

CalStandard

CalManager

CalibrateAllChannels

CalSets

CalSet

GuidedCalibration

GuidedCalibration
PowerSensors

GuidedCalibration
PowerSensor

ECalUserCharacterizer

ECalModules

ECalModule

PhaseReferenceCal

SMCType

VMCType

NoiseCal

GainCompressionCal

SweptIMDCal

SourcePowerCalibrator

PowerLossSegments

PowerLossSegment

PowerMeterInterfaces

PowerMeterInterface

PowerSensors

PowerSensor

ExternalSource

PowerSensorAsReceiver

PowerLossSegmentsPMAR

PowerLossSegmentPMAR

CalFactorSegsPMAR

PowerSensorCal
FactorSegmentPMAR

ExternalTestSets

TestSetControl

FIFO

GlobalPowerLimit

HWAUXIO

HWExternalTestSet

HWMaterialHandlerIO

InterfaceControl

IOConfiguration

PathConfigurationMgr

PortExtension

Preferences

Display/PrintColors

Trace(1-8)

SCPIStringParser

TriggerSetup

CalFactorSegs

**PowerSensorCal
FactorSegment**

Legend:

Object

Collection

Interface

Last Modified:

18-Sep-2012 Added CalAll and Phase Ref
21-Feb-2012 Added DCStimulus, ExtDCDevice, ExtPulse
23-May-2011 Added MeasurementClassProperties
1-Dec-2010 Added GuidedPowerSensor(s) and PhaseControl
10-Sep-2010 Added CorrectionMethods
29-Jun-2010 Fixed PathElement object
19-Feb-2010 Added PSAT, PNOP, GDAperture (9.2)
27-Aug-2009 Added External Devices
7-Apr-2009 Added ECalModules/Module
13-Nov-2007 Replaced image with text

Added NFA, ENRFile, and GCA

Application Object

Description

The Application object is the highest object in the PNA [object model](#). This object presents methods and properties that affect the entire analyzer, rather than a specific channel or measurement. For example, the application object provides the GetIDString method. There's only one ID string for the instrument, unrelated to the channel or parameter being measured. Likewise, the TriggerSignal Property is global to the instrument. You can elect to use an internally generated (free run) trigger or a manual trigger. Either way, that type of trigger generation will be used on all measurements, on all channels. Therefore, it is under the Application object.

Accessing the Application object

This object is unique in that you must **create** this object rather than just get a handle to it.

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)
```

Replace <analyzerName> with the full computer name of your PNA. For example, "My PNA". See [Change Computer Name](#).

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Getting a Handle to an Object](#).
- [Example Programs](#)
- [Superseded commands](#)

(**Bold** Methods or Properties provide access to a child object)

Methods	Interface	Description
	See History	
ActivateWindow	IApplication	Makes a window object the Active Window.
AllowAllEvents	IApplication	Monitors all events
AllowEventCategory	IApplication	Monitors an event category
AllowEventMessage	IApplication	Monitors an event

AllowEventSeverity	IApplication	Monitors an event severity level
BuildHybridKit	IApplication	Defines the user kit as port1kit + port2kit.
Channel	IApplication	Returns a handle to the channel object for the supplied channel number.
Configure	IApplication9	Restarts as an "N-port" PNA using the specified multiport test set.
CreateCustomMeasurementEx	IApplication3	Creates a new custom measurement with initialization.
CreateCustomMeasurement	IApplication	Superseded with CreateCustomMeasurementEx Method
CreateMeasurement	IApplication	Creates a new measurement.
CreateSParameter	IApplication	Creates a new S-Parameter measurement.
CreateSParameterEx	IApplication	Superseded with Create SParameter Method
DeleteShortcut	IApplication	Removes a macro (shortcut) from the list of macros
DisallowAllEvents	IApplication	Monitors NO events
DoPrint	IApplication	Prints the screen to the active Printer.
ExecuteShortcut	IApplication	Executes a macro (shortcut) stored in the analyzer.
GetAuxIO	IApplication	Returns a handle to the AuxIO interface
GetCalManager	IApplication	Returns a handle to the CalManager interface
GetExternalTestSetIO	IApplication	Returns a handle to the ExternalTestSet IO interface
GetIPConfigurationStruct	IApplication14	Returns an NA_IPConfiguration data structure which contains information about the current status of the PNA computer networking configuration.
GetMaterialHandlerIO	IApplication	Returns a handle to the MaterialHandlerIO interface
GetShortcut	IApplication	Returns the title and path of the specified macro (shortcut).
LANConfigurationInitialize	IApplication13	Resets the PNA LAN configuration.
LaunchCalWizard	IApplication	Launches the Cal Wizard
LaunchDialog	IApplication10	Launches the specified dialog box.
ManualTrigger	IApplication	Triggers the analyzer when TriggerSignal = naTriggerManual.
Preset	IApplication	Resets the analyzer to factory defined default settings.

PrintToFile	IApplication	Saves the screen data to bitmap (.bmp) file of the screen.
PutShortcut	IApplication	Puts a Macro (shortcut) file into the analyzer.
Quit	IApplication	Ends the Network Analyzer application.
Recall	IApplication	Recalls a measurement state, calibration state, or both from the hard drive into the analyzer.
RecallKits	IApplication	Recalls the calibration kits definitions that were stored with the SaveKits command.
Reset	IApplication	Removes all existing windows and measurements.
RestoreCalKitDefaults	IApplication	Restores the factory defaults for the specified kit.
RestoreCalKitDefaultsAll	IApplication	Restores the factory defaults for all kits.
Save	IApplication	Saves instrument state and calibration files to disk
SaveCitiDataData	IApplication5	Saves UNFORMATTED trace data to .cti file. Superseded with SaveData
SaveCitiFormattedData	IApplication5	Saves FORMATTED trace data to .cti file. Superseded with SaveData
SaveData	IApplication18	Saves trace data to files on disk.
SaveKits	IApplication	Saves all cal kits to disk.
SetFailOnOverRange	IApplication	Causes over range values to return an error code
SetIPConfiguration	IApplication14	Modifies settings of the PNAs computer networking configuration.
ShowStatusBar	IApplication	Shows and Hides the Status Bar.
ShowStimulus	IApplication	Shows and Hides Stimulus information.
ShowTitleBars	IApplication	Shows and Hides the Title Bars.
ShowToolbar	IApplication	Shows and Hides the specified Toolbar.
UserPreset	IApplication7	Performs a User Preset.
UserPresetLoadFile	IApplication7	Loads an existing instrument state file (.sta or .cst) to be used for User Preset.
UserPresetSaveState	IApplication7	Saves the current instrument settings as UserPreset.sta.

Properties

Description

<u>ActiveCalKit</u>	IApplication	Returns a pointer to the kit identified by kitNumber.
<u>ActiveChannel</u>	IApplication	Returns a handle to the Active Channel object.
<u>ActiveMeasurement</u>	IApplication	Returns a handle to the Active Measurement object.
<u>ActiveNAWindow</u>	IApplication	Returns a handle to the Active Window object.
<u>ArrangeWindows</u>	IApplication	Sets or returns the arrangement of all the windows.
<u>AuxiliaryTriggerCount</u>	IApplication11	Returns the number of Aux trigger input / output connector pairs in the instrument.
<u>CalKitType</u>	IApplication	Sets or returns the calibration kit type for to be used for calibration or for kit modification. Shared with the CalKit object.
<u>Capabilities</u>	IApplication4	Return capabilities of the remote PNA.
<u>Channels</u>	IApplication	Collection for iterating through the channels
<u>CoupledMarkers</u>	IApplication	Sets (or reads) coupled markers ON and OFF
<u>DisplayAutomationErrors</u>	IApplication2	Enables or disables automation error messages from being displayed on the screen. U
<u>DisplayGlobalPassFail</u>	IApplication6	Shows or hides the dialog which displays global pass/fail results.
<u>E5091Testsets</u>	IApplication8	Collection to control the E5091A testset.
<u>ENRFile</u>	IApplication13	Manages Noise ENR files.
<u>ExternalALC</u>	IApplication	Sets or returns the source of the analyzer leveling control.
<u>ExternalDevices</u>	IApplication16	Collection to control External Devices
<u>ExternalTestsets</u>	IApplication9	Collection to control External Test sets.
<u>FIFO</u>	IApplication15	Controls FIFO settings
<u>GlobalPowerLimit</u>	IApplication17	Controls Global Power Limit settings
<u>GPIBAddress</u>	IApplication8	Sets and returns the PNA GPIB address.
<u>GPIBMode</u>	IApplication	Makes the analyzer the system controller or a talker/listener.
<u>GridLineType</u>	IApplication17	Set and return the line type of the window grid (solid dotted)
<u>IDString</u>	IApplication	Returns the model, serial number and software revision of the analyzer

<u>InterfaceControl</u>	IApplication8	Control the Interface control features.
<u>IOConfiguration</u>	IApplication19	Provides access to IO Configuration commands
<u>LANConfiguration</u>	IApplication13	Returns the current status of the PNA computer networking configuration.
<u>LXIDeviceIDState</u>	IApplication14	Displays the LAN Status dialog with LAN Status Indicator showing IDENTIFY.
<u>LocalLockoutState</u>	IApplication4	Prevents use of the mouse, keyboard, and front panel while your program is running.
<u>Measurement</u>	IApplication	Create and manage measurements
<u>Measurements</u>	IApplication	Collection for iterating through the Application measurements.
<u>MessageText</u>	IApplication	Returns text for the specified eventID
<u>NaWindows</u>	IApplication	Collection for iterating through the Application windows.
<u>NoiseSourceState</u>	IApplication13	Sets and Reads the ON OFF state of the noise source
<u>NumberOfPorts</u>	IApplication	Returns the number of hardware source ports on the PNA
<u>Options</u>	IApplication	Returns the options on the analyzer
<u>PathConfigurationManager</u>	IApplication11	Provides access to hardware configuration.
<u>Port Extensions</u>	IApplication	Superseded with <u>Fixturing Object</u>
<u>Preferences</u>	IApplication5	Preferences for many PNA settings..
<u>ScpiStringParser</u>	IApplication	Provides the ability to send a SCPI command from within the COM command.
<u>SecurityLevel</u>	IApplication4	Turns ON or OFF the display of frequency information.
<u>SICL</u>	IApplication5	Allows control of the PNA via SICL
<u>SICLAddress</u>	IApplication8	Sets and returns the PNA SICL address
<u>SourcePowerCalibrator</u>	IApplication2	Allows capability for performing source power calibrations.
<u>SourcePowerState</u>	IApplication	Turns Source Power ON and OFF.
<u>SystemImpedanceZ0</u>	IApplication	Sets the analyzer impedance value.
<u>SystemName</u>	IApplication	Returns the full computer name of the PNA.

Touchscreen	IApplication12	Enables and disables touchscreen.
TriggerDelay	IApplication	Sets or returns the delay time for a trigger.
TriggerSetup	IApplication4	Controls triggering for the entire PNA application.
TriggerSignal	IApplication	Superseded with Source Property
TriggerType	IApplication	Superseded with Scope Property
UserPresetEnable	IApplication7	'Checks' and 'clears' the enable box on the User Preset dialog box.
VelocityFactor	IApplication	Sets the velocity factor to be used with Electrical Delay, Port Extensions, and Time Domain marker distance calculations.
Visible	IApplication	Makes the Network Analyzer application visible or not visible.
WindowState	IApplication	Sets or returns the window setting of Maximized, Minimized, or Normal. Shared with the NAWindow Object

Events	Interface	Description
OnCalEvent	IApplication	Triggered by a calibration event.
OnChannelEvent	IApplication	Triggered by a channel event.
OnDisplayEvent	IApplication	Triggered by a display event.
OnHardwareEvent	IApplication	Triggered by a hardware event.
OnMeasurementEvent	IApplication	Triggered by a measurement event.
OnSCPIEvent	IApplication	Triggered by a SCPI event.
OnSystemEvent	IApplication	Triggered by a system event.
OnUserEvent	IApplication	For future use

IApplication History

Interface	Introduced with PNA Rev:
IApplication	1.0
IApplication2	3.0
IApplication3	3.2
IApplication4	3.5
IApplication5	4.0
IApplication6	5.0
IApplication7	5.0
IApplication8	5.2
IApplication9	6.0
IApplication10	7.20
IApplication11	7.20
IApplication12	7.21
IApplication13	8.0
IApplication14	8.2
IApplication15	8.34
IApplication16	9.0
IApplication17	9.0
IApplication18	9.2
IApplication19	10.0

Last Modified:

17-Oct-2007 Updated IPathConfigMgr Prop

AuxiliaryTrigger Object

Description

These properties setup Auxiliary triggering on a channel.

Accessing the object

Use `chan.AuxTrigger (n)` to access the object.

where **n**= the connector pair to be used for Auxiliary Triggering.

- PNA-X models: Use **1** or **2**
- All other PNA models: Use **1** - **These models do NOT have an Aux Input; only an Output. Therefore, the following 'Input' commands will return an error when sent to PNA models other than the PNA-X.**

Use [app.AuxiliaryTriggerCount](#) to determine the number of auxiliary trigger pairs on the rear panel of a PNA.

```
Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel
Dim AuxTrig as AuxTrigger
AuxTrig = chan.AuxTrigger(2)
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Triggering in the PNA](#)
- [Example Programs](#)

Methods	Interface	Description
	See History (below)	

None

Properties		Description
Delay (Input)	IAuxTrigger	Specifies the delay that should be applied by the PNA after the Aux trigger input is received and before the acquisition is made
Enable	IAuxTrigger	Turns ON / OFF the trigger output.
HandshakeEnable (Input)	IAuxTrigger	Turns handshake ON / OFF.
Number	IAuxTrigger	Reads the number of the Aux I/O pair being used.
TriggerInPolarity (Input)	IAuxTrigger	Specifies the polarity of the trigger IN signal to which the PNA will respond.
TriggerInType (Input)	IAuxTrigger	Specifies the type of Aux trigger input being supplied to the PNA
TriggerOutDuration	IAuxTrigger	Specifies the width of the pulse or the time that the Aux trigger output will be asserted
TriggerOutInterval	IAuxTrigger	Specifies how often a trigger output signal is sent.
TriggerOutPolarity	IAuxTrigger	Specifies the polarity of the trigger output signal being supplied by the PNA.
TriggerOutPosition	IAuxTrigger	Specifies whether the Aux trigger out signal is sent Before or After the acquisition.

IAuxTrigger History

Interface	Introduced with PNA Rev:
IAuxTrigger	7.2

Last Modified:

6-Apr-2009 Replaced N5242A with PNA-X
13-Aug-2008 Fixed example

BalancedMeasurement Object

Description

These properties set the measurement type that is used with balanced topologies.

Use the [BalancedTopology Object](#) to set the topology and port mappings for the DUT,

Accessing the BalancedMeasurement object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim balMeas As BalancedMeasurement
Set balMeas = app.ActiveMeasurement.BalancedMeasurement
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [About Balanced Measurements](#)
- [Example Programs](#)

(**Bold** Methods or Properties provide access to a child object)

Method	Description	
None		
Property	Interface	Description
	See History	
BalancedMode	IBalancedMeasurement	Sets and returns whether the balanced transform is ON or OFF.
BalancedStimulus	IBalancedMeasurement2	Sets and returns the stimulus properties of a balanced DUT.
BalancedTopology	IBalancedMeasurement	Sets and returns the topology of a balanced DUT.

BalSMeasurement	IBalancedMeasurement3	Sets and returns the measurement for the Balanced - Single-ended topology.
BBalMeasurement	IBalancedMeasurement	Sets and returns the measurement for the Balanced - Balanced topology.
SBalMeasurement	IBalancedMeasurement	Sets and returns the measurement for the Single-Ended - Balanced topology.
SSBMeasurement	IBalancedMeasurement	Sets and returns the measurement for the Single-Ended - Single-Ended - Balanced topology

IBalancedMeasurement History

Interface	Introduced with PNA Rev:
IBalancedMeasurement	5.0
IBalancedMeasurement2	8.2
IBalancedMeasurement3	9.70

BalancedStimulus Object

Description

These properties set the values that are unique to iTMSA - Opt 460.

All other properties for iTMSA use the standard PNA commands.

Accessing the BalancedStimulus object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim balStim As BalancedStimulus
Set balStim = app.ActiveMeasurement.BalancedMeasurement.BalancedStimulus
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [About iTMSA](#)
- [Example iTMSA Program](#)

(**Bold** Methods or Properties provide access to a child object)

Method	Description
None	

Property	Interface	Description
	See History	
BalPort1PhaseOffset	IBalancedStimulus	Sets balanced port 1 phase offset
BalPort1PowerOffset	IBalancedStimulus	Sets balanced port 1 power offset
BalPort1StartPhase	IBalancedStimulus2	Sets Phase start value for port 1
BalPort1StopPhase	IBalancedStimulus2	Sets Phase stop value for port 1
BalPort2PhaseOffset	IBalancedStimulus	Sets balanced port 2 phase offset
BalPort2PowerOffset	IBalancedStimulus	Sets balanced port 2 power offset
BalPort2StartPhase	IBalancedStimulus2	Sets Phase start value for port 2

BalPort2StopPhase	IBalancedStimulus2	Sets Phase start value for port 2
Mode	IBalancedStimulus	Sets Stimulus mode for balanced measurements
PhaseAsFixture	IBalancedStimulus	Sets the state of phase offset as a fixture
PhaseSwpAsFixture	IBalancedStimulus2	Enable Phase Sweep as fixture
PhaseSwpState	IBalancedStimulus2	Enable Phase Sweep
PowerAsFixture	IBalancedStimulus	Sets the state of power offset as a fixture

IBalancedStimulus History

Interface	Introduced with PNA Rev:
IBalancedStimulus	8.2
IBalancedStimulus2	8.5

Last Modified:

27-Feb-2009	Added IBal2 - Phase Sweep
15-May-2008	MX New topic

BalancedTopology Object

Description

The [DUTTopology](#) property sets and returns the topology of a balanced DUT.

The following methods **set** the port mappings for the DUT.

The remaining properties **return** the port mappings for the DUT.

Use the [BalancedMeasurement object](#) to set the measurement type.

Accessing the BalancedTopology object

```
Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel

Dim balTopology as BalancedTopology
Set balTopology = chan.BalancedTopology
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [About Balanced Measurements](#)
- [Example Programs](#)

Method	Interface	Description
	See History	
SetBBPorts	IBalancedTopology	Sets the physical port mappings for the Balanced - Balanced DUT topology.
SetBSPorts	IBalancedTopology2	Sets the physical port mappings for the Balanced - Single-Ended DUT topology.
SetSBPorts	IBalancedTopology	Sets the physical port mappings for the Single-Ended - Balanced DUT topology.
SetSSBPorts	IBalancedTopology	Sets the physical port mappings for the Single-Ended - Single-Ended - Balanced DUT topology.

Property	Interface	Description
<u>BB_BalPort1Negative</u>	IBalancedTopology	Returns the PNA port number that is connected to the Negative side of the DUT's logical Port 1 .
<u>BB_BalPort1Positive</u>	IBalancedTopology	Returns the first positive balanced port number in the Balanced - Balanced topology
<u>BB_BalPort2Negative</u>	IBalancedTopology	Returns the second negative balanced port number in the Balanced - Balanced topology.
<u>BB_BalPort2Positive</u>	IBalancedTopology	Returns the second positive balanced port number in the Balanced - Balanced topology.
<u>BS_BalPortNegative</u>	IBalancedTopology2	Returns the negative balanced port number in the Balanced - Single-ended topology.
<u>BS_BalPortPositive</u>	IBalancedTopology2	Returns the positive balanced port number in the Balanced - Single-ended topology.
<u>BS_SEPort</u>	IBalancedTopology2	Returns the single-ended port number in the Balanced - Single-ended topology.
<u>DUTTopology</u>	IBalancedTopology	Sets and returns the device topology setting.
<u>SB_BalPortNegative</u>	IBalancedTopology	Returns the negative balanced port number in the Single-Ended - Balanced topology.
<u>SB_BalPortPositive</u>	IBalancedTopology	Returns the positive balanced port number in the Single-Ended - Balanced topology.
<u>SB_SEPort</u>	IBalancedTopology	Returns the single ended port number in the Single-Ended - Balanced topology.
<u>SSB_BalPortNegative</u>	IBalancedTopology	Returns the negative balanced port number in the Single-Ended - Single-Ended - Balanced topology.
<u>SSB_BalPortPositive</u>	IBalancedTopology	Returns the positive balanced port number in the Single-Ended - Single-Ended - Balanced topology
<u>SSB_SEPort1</u>	IBalancedTopology	Returns the first single ended port in the Single-Ended - Single-Ended - Balanced topology.
<u>SSB_SEPort2</u>	IBalancedTopology	Returns the second single ended port in the Single-Ended - Single-Ended - Balanced topology.

BalancedTopology History

Interface	Introduced with PNA Rev:
IBalancedTopology	5.0
IBalancedTopology2	9.70

CalFactorSegments Collection

Description

A collection object that provides a mechanism for iterating through the segments of a power sensor cal factor table. The Cal Factor table can contain up to 100 segments.

Accessing the CalFactorSegments collection

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim calFact As CalFactorSegments
Set calFact = app.SourcePowerCalibrator.PowerSensors(1).CalFactorSegments
```

See Also:

- [PowerSensorCalFactorSegment Object](#)
- [About Source Power Cal](#)
- [Collections in the Analyzer](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods	Description
Add	Adds a PowerSensorCalFactorSegment object to the collection
Item	Use to get a handle to a PowerSensorCalFactorSegment object in the collection.
Remove	Removes an object from the collection.
Properties	Description
Count	Returns the number of objects in the collection.
Parent	Returns a handle to the Parent object (PowerSensor) of this collection.

CalFactorSegmentsPMAR Collection

Description

A collection object that provides a mechanism for iterating through the segments of a power sensor cal factor table. The Cal Factor table can contain up to 100 segments. This collection is used when the Power Meter is used as a Receiver.

Accessing the CalFactorSegmentsPMAR collection

Example: [Create a PMAR Device and Measurement](#)

See Also:

- [PowerSensorCalFactorSegmentPMAR Object](#)
- [About PMAR](#)
- [Collections in the Analyzer](#)
- [The PNA Object Model](#)

Methods	Description
Add	Adds a PowerSensorCalFactorSegmentPMAR object to the collection
Item	Use to get a handle to a PowerSensorCalFactorSegmentPMAR object in the collection.
Remove	Removes an PowerSensorCalFactorSegmentPMAR object from the collection.

Properties	Description
Count	Returns the number of PowerSensorCalFactorSegmentPMAR objects in the collection.
Parent	Returns a handle to the Parent object of this collection.

Last Modified:

25-Aug-2009 MX New topic

CalibrateAllChannels Object

Description

Use this interface to Calibrate All Channels.

See Cal All Channels limitations .

Accessing the CalibrateAllChannels Object

```
Dim app as AgilentPNA835x.Application

Dim CalMgr

Set CalMgr = app.GetCalManager

Dim calAll

Set calAll = CalMgr.CalibrateAllChannels

'Then get a handle to the GuidedCal object

Dim guidedCal

Set guidedCal = CalAll.GuidedCalibration
```

See Also:

- Example program
- Learn about Calibrate All Channels .
- PNA Automation Interfaces
- The PNA Object Model
- **Superseded** commands

(Bold Methods or Properties provide access to a child object)

In the following table ICalibrateAllChannels is abbreviated to ICalAllChans

Methods	Interface	Description
	See History	
Reset	ICalAllChans	Resets all properties associated with the Cal All session to their default values.

Properties

CalibrationPorts	ICalAllChans	For each channel, sets and returns the ports to be calibrated.
Channels	ICalAllChans	Sets and returns the list of channels to be calibrated during the Cal All session.
GeneratedCalsets	ICalAllChans	Returns the cal set names that were produced by the cal all session.
GuidedCalibration	ICalAllChans	Provides access to the GuidedCal object. Use this to perform the Calibration.
IFBW	ICalAllChans	Sets and returns the IFBW for a Cal All calibration
PathConfigurationElement	ICalAllChans	Sets and returns the Path Configuration settings for a Cal All calibration.
PowerLevel	ICalAllChans	Sets and returns the power level at which a Cal All calibration is to be performed.
PowerOffset	ICalAllChans	Sets and returns the power offset value for a Cal All calibration.
PropertyNames	ICalAllChans	Returns the settable properties for the current cal all session
PropertyValue	ICalAllChans	Sets and returns a value for a specific property name
PropertyValues	ICalAllChans	Returns the valid property values for a specific property name:
ReceiverAttenuator	ICalAllChans	Sets and returns the Receiver Attenuator setting for a Cal All calibration.
SourceAttenuator	ICalAllChans	Sets and returns the Source Attenuator setting for a Cal All calibration.
SParameterCalPorts	ICalAllChans	Returns a list of ports to be calibrated.
UserCalsetPrefix	ICalAllChans	Sets and returns the prefix to be used when saving User CalSets that result from the Cal All session.

ICalibrateAllChannels History

Interface	Introduced with PNA Rev:
------------------	---------------------------------

ICalibrateAllChannels	9.50
-----------------------	------

Last modified:

3-Jan-2012	New topic
------------	-----------

Calibrator Object

See Also

- [Example Programs](#)
- [Calibrator Methods and Properties](#)
- [ICalData Interface](#) for putting and getting typed Calibration data.
- [Superseded commands](#)

Description

The Calibrator object, a child of the channel, is used to perform an **Unguided** calibration.

Important!

Do **NOT** use commands from the [GuidedCalibration](#) object when performing an Unguided calibration. Use **ONLY** the Calibrator object.

You can NOT perform a full 3 or 4-port using the Calibrator object. You must use the [GuidedCalibration object](#).

There must be a measurement present for the calibrator to use or you will receive a "no measurement found" error. Therefore, to perform a 2-port cal, you must have any S-parameter measurement on the channel. For a 1-port measurement, you must have the measurement (S11 or S22) on the channel. The same is true for a response measurement.

There are a number of approaches to calibration with the calibrator object:

- You can collect data yourself and download it to the ACQUISITION buffer. The acquisition buffer holds the actual measured data for each standard. See the PNA [data map](#).
 1. Calibrator.[SetCallInfo](#)
 2. Connect a standard
 3. Trigger a sweep
 4. Retrieve the data for the standard
 5. Download the data - calibrator.[putStandard](#)

6. Repeat for each standard
 7. Calibrator.[CalculateErrorCoefficients](#)
- You can tell the calibrator to acquire a standard. In this case, the calibrator collects the data and places it in the ACQUISITION buffer.
 1. Calibrator.[SetCallInfo](#)
 2. Connect a standard
 3. Calibrator.[AcquireCalStandard2](#)
 4. Repeat for each standard
 5. Calibrator.[CalculateErrorCoefficients](#)
 - You can put previously-retrieved error terms in the error correction buffer.
 1. [PutErrorTerm](#)
 2. Repeat for each term
 3. Measurement.[Caltype](#) = pick one
 - You can also "piece together" a 2-port cal from two 1-port calcs (S11 and S22) and four response (thru) calcs. The system will detect that all the standards needed for a 2-port cal have been acquired even though they may not have gathered at the same time.

Accessing the Calibrator object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim cal As ICalibrator
Set cal = app.ActiveChannel.Calibrator
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Learn about reading and writing Calibration data.](#)

Methods	Interface	Description
		See History
AcquireCalConfidenceCheckECAL	ICalibrator	Superseded with AcquireCalConfidenceCheckECALEx
AcquireCalConfidenceCheckECALEx	ICalibrator4	Transfers ECAL confidence data into analyzer memory
AcquireCalStandard	ICalibrator	Superseded with AcquireCalStandard2
AcquireCalStandard2	ICalibrator	Causes the analyzer to measure a calibration standard. Also provides for sliding load.
CalculateErrorCoefficients	ICalibrator	Generates Error Terms from standard and actual data in the error correction buffer.
DoECAL1Port	ICalibrator	Superseded with DoECAL1PortEx
DoECAL1PortEx	ICalibrator4	Completes a 1 port ECAL
DoECAL2Port	ICalibrator	Superseded with DoECAL2PortEx
DoECAL2PortEx	ICalibrator4	Completes a 2 port ECAL
DoneCalConfidenceCheckECAL	ICalibrator	Concludes an ECAL confidence check
DoReceiverPowerCal	ICalibrator5	Perform a receiver power cal.
DoResponseCal	ICalibrator9	Perform a response (normalization) cal.
GetCalKitTypeString	ICalibrator8	Returns information about the attached modules
GetECALModuleInfo	ICalibrator	Superseded with GetECALModuleInfoEx
GetECALModuleInfoEx	ICalibrator4	Returns information about the attached module
getErrorTerm	ICalibrator	Superseded with GetErrorTermByString
getStandard	ICalibrator	Superseded with GetStandardByString
putErrorTerm	ICalibrator	Superseded with PutErrorTermByString
putStandard	ICalibrator	Superseded with PutStandardByString
SaveCalSets	ICalibrator	Superseded with CalSet.Save
setCalInfo	ICalibrator	Specifies the type of calibration and prepares the internal state for the rest of the calibration.
Properties	Interface	Description

AcquisitionDirection	ICalibrator	Specifies the direction in a 2-Port cal using one set of standards.
CalKitType	ICalibrator10	Sets and returns the name of the Cal Kit to use for unguided cal.
CalKitTypes	ICalibrator10	Returns the names of the first 50 mechanical cal kits in your PNA that can be used for unguided calibrations.
ECALCharacterization	ICalibrator2	Superseded with ECALCharacterizationEx
ECALCharacterizationEx	ICalibrator4	Specifies which set of characterization data within an ECal module will be used for ECal operations with that module.
ECALCharacterizationIndexList	ICalibrator6	Returns a list of characterizations stored in the specified ECal module.
ECAL Isolation	ICalibrator	Specifies whether the acquisition of the ECal calibration should include isolation or not.
ECALModuleNumberList	ICalibrator6	Returns a list of index numbers to be used for referring to the ECal modules that are currently attached to the PNA.
ECALPortMap	ICalibrator3	Superseded with ECALPortMapEx
ECALPortMapEx	ICalibrator4	Specifies which ports of the ECal module are connected to which ports of the PNA.
IsECALModuleFound	ICalibrator	Superseded with IsECALModuleFoundEx
IsECALModuleFoundEx	ICalibrator4	Superseded with ECALCharacterizationIndexList and ECALModuleNumberList
IsolationAveragingIncrement	ICalibrator7	Value to increase the channel's averaging factor.
OrientECALModule	ICalibrator3	Specifies if the PNA should perform orientation of the ECal module during calibration.
Simultaneous2PortAcquisition	ICalibrator	Allows the use of 2 sets of standards at the same time.

ICalibrator History

Interface	Introduced with PNA Rev:
ICalibrator	1.0
ICalibrator2	3.1
ICalibrator3	3.1
ICalibrator4	3.5
ICalibrator5	5.0
ICalibrator6	5.26
ICalibrator7	7.21
ICalibrator8	8.1
ICalibrator9	9.1
ICalibrator10	9.2

ICalData Interface

Description

Contains methods for putting Calibration data in and getting Calibration data out of the analyzer using typed data. This interface transfers data more efficiently than variant data. However, this interfaces is only usable from VB6, C, & C++. All other programming languages must use the [ICalSet interface](#).

There is also an [ICalDataInterface](#) on the CalSet Object

[Learn about reading and writing Calibration data.](#)

Methods	Description
getErrorTermComplex	Retrieves error term data
getStandardComplex	Retrieves calibration data from the acquisition data buffer (before error-terms are applied).
putErrorTermComplex	Puts error term data
putStandardComplex	Puts calibration data into the acquisition data buffer (before error-terms are applied).
Properties	Description
None	

ICalData History

Interface **Introduced with PNA**
Rev:

ICalData 1.0

CalKit Object

Description

The calkit object provides the properties and methods to access and modify a calibration kit.

Accessing a CalKit object

The **active** cal kit is the kit that is selected for use in **Unguided** calibrations. To get a handle to the active kit, use the [app.ActiveCalKit](#) property. To access the CalKit object for a specific cal kit, you must first make that kit the active kit using [app.CalKitType](#).

The CalKit object behaves differently from other objects in that you can only have a handle to **one** cal kit -- the active cal kit. Therefore, when you change the CalKitType from either the [Application object](#) or the [CalKit object](#), you may also be changing the object to which you may have other references.

For example, the following example specifies two CalKit type objects and in turn, assigns them to two different variables: ck1 and ck2.

```
Dim app As AgilentPNA835x.Application
Dim ck1 As calKit
Dim ck2 As calKit

Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)
app.CalKitType = naCalKit_User1
Set ck1 = app.ActiveCalKit
ck1.Name = "My CalKit1"

app.CalKitType = naCalKit_User2
Set ck2 = app.ActiveCalKit
ck2.Name = "My CalKit2"

Print "ck1: " & ck1.Name
Print "ck2: " & ck2.Name
```

When the pointer to each of these kits is read (printed), they each have a pointer to the last kit to be assigned to the active cal kit:

```
ck1: My CalKit2
ck2: My CalKit2
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)

- [Example Programs](#)

(**Bold** Methods or Properties provide access to a child object)

Methods	Description
getCalStandard	Returns a handle to a calibration standard for modifying its definitions.
GetStandardsForClass	Returns the calibration standard numbers for a specified calibration class.
SetStandardsForClass	Sets the calibration standard numbers for a specified calibration class

Properties	Description
CalKitType	Sets or returns the calibration kit type to be used for calibration or for kit modification. Shared with the Application object.
Name	Sets and returns the name of the cal kit
PortLabel	Labels the ports for the kit; only affects the cal wizard annotation.
StandardForClass	Superseded with Use GetStandardForClass and SetStandardForClass . Maps a standard device to a cal class.

ICalKit History

Interface	Introduced with PNA Rev:
ICalKit	1.0

CalManager Object

Description

Use this interface to list, save, and delete Cal Sets.

Accessing the CalManager object

Get a handle to a the CalManager with the app.GetCalManager Method.

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim mgr as ICalManager
Set mgr = app.GetCalManager
```

See Also:

- PNA Automation Interfaces
- The PNA Object Model
- Example Programs
- **Superseded** commands

(**Bold** Methods or Properties provide access to a child object)

Methods	Interface	Description
	See History	
AllowChannelToSweepDuringCalAcquisition	ICalManager5	Specifies the channel to sweep during a Calibration.
CreateCalSet	ICalManager	Creates a new Cal Set
CreateCustomCal	ICalManager2	Creates an FCA cal object.
CreateCustomCalEx	ICalManager5	Creates a custom cal object.

Deembed	ICalManager8	De-embeds a fixture from an existing Cal Set based on an S2P file.
DeleteCalSet	ICalManager	Deletes a Cal Set
DisplayNAWindowDuringCalAcquisition	ICalManager5	Set the 'show' state of the window to be displayed during a calibration.
DisplayOnlyCalWindowDuringCalAcquisition	ICalManager5	Clears the flags for windows to be shown during calibrations.
Embed	ICalManager8	Embeds a fixture into an existing Cal Set based on an S2P file.
EnumerateCalSets	ICalManager4	Returns an array of Cal Set names being stored on the PNA.
GetCalSetByGUID	ICalManager	Get a handle to a Cal Set
GetCalSetCatalog	ICalManager	Superseded with EnumerateCalSets
GetCalSetUsageInfo	ICalManager	Returns the Cal Set ID and Error Term ID currently in use
GetCalTypes	ICalManager2	Query for a list of available calibration types.
GetEcalUserCharacterizer	ICalManager6	Returns the ECalUserCharacterizer object.
GetRequiredEtermNames	ICalManager2	Returns an array of strings specifying the error terms required by the Cal Type correction algorithm.

SaveCalSets	ICalManager	Superseded with CalSet.Save
-------------	-------------	---------------------------------------

SweepOnlyCalChannelDuringCalAcquisition	ICalManager5	Clears ALL flags for channels to sweep during calibration.
---	--------------	--

Properties

CalSets Collection	ICalManager	Collection for iterating through all the Cal Sets in the analyzer.
---------------------------	-------------	--

ECalModules Collection	ICalManager7	Collection of ECal Modules that are connected to the PNA.
-------------------------------	--------------	---

GuidedCalibration	ICalManager3	Used to perform a Guided Calibration.
--------------------------	--------------	---------------------------------------

ICalManager History

Interface	Introduced with PNA Rev:
ICalManager	2.0
CalManager2	3.1
CalManager3	3.5
CalManager4	5.0
ICalManager5	8.0
ICalManager6	8.3
ICalManager7	8.5
ICalManager8	9.33

CalSet Object

See [ICalData Interface](#) for putting and getting typed Cal Set data.

Description

Use this interface to query and or change the contents of a Cal Set.

Accessing the CalSet object

Get a handle to a CalSet object by using the CalSets collection. This is done through the CalManager object with the app.[GetCalManager](#) Method.

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim calst As ICalSet
Set calst = app.GetCalManager.CalSets.Item(1)
'OR Get a handle by CalSet Name
Set calst = app.GetCalManager.CalSets.Item("MyCalSet")
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Reading and Writing Calibration data](#)
- [Example Programs](#)
- [Superseded commands](#)

Methods	Interface	Description
	See History	
CloseCalSet	ICalSet	Obsolete - No longer necessary.
ComputeErrorTerms	ICalSet	Computes error terms for the CalType specified by a preceding OpenCal Set call.
Copy	ICalSet	Creates a new Cal Set and copies the current Cal Set data into it.
EnumerateItems	ICalSet6	Returns a list of all name-value pairs (items) in the Cal Set.

getErrorTerm	ICalSet	Superseded with getErrorTermByString
getErrorTermByString	ICalSet2	Returns variant error term data by specifying the string name of the error term.
getErrorTermList	ICalSet	Superseded with getErrorTermList2
getErrorTermList2	ICalSet2	Returns a list of error term names found in a calset.
GetErrorTermStimulus	ICalSet7	Returns the stimulus values over which the specific error term was acquired.
GetGUID	ICalSet	Returns the GUID identifying a Cal Set
getStandard	ICalSet	Superseded with getStandardByString
getStandardByString	ICalSet2	Returns variant standard acquisition data by specifying the string name of the standard.
getStandardsList	ICalSet	Superseded with getStandardList2
getStandardList2	ICalSet2	Returns a list of standard names found in a Cal Set.
HasCalType	ICalSet	Verifies that the Cal Set object contains the error terms required to apply the specified CalType to an appropriate measurement.
OpenCalSet	ICalSet	Obsolete - No longer necessary.
putErrorTerm	ICalSet	Superseded with putErrorTermByString
putErrorTermByString	ICalSet2	Writes variant error term data by specifying the string name of the error term.
PutErrorTermStimulus	ICalSet7	Adds stimulus data to the specified buffer.
putStandard	ICalSet	Superseded with putStandardByString
putStandardByString	ICalSet2	Writes variant standard acquisition data by specifying the string name of the standard.
RemoveItem	ICalSet6	Removes a name-value pair from the Cal Set.
Save	ICalSet	Saves the current Cal Set to disk.
StringToNACalClass	ICalSet	Converts string values from GetStandardsList into enumeration data
StringToNAErrorTerm2	ICalSet	Converts string values from GetErrorTermList into enumeration data
Properties		Description

<u>AlternateSweep</u>	ICalSet3	Reads sweep either alternate or chopped.
<u>Attenuator</u>	ICalSet3	Returns the value of the attenuator control for the specified port number.
<u>AttenuatorMode</u>	ICalSet3	Returns the mode of operation (auto or manual) of the attenuator control for the specified port number.
<u>CouplePorts</u>	ICalSet3	Returns state of couple ports (ON or OFF)
<u>CWFrequency</u>	ICalSet3	Returns CW Frequency
<u>Description</u>	ICalSet	Set or return the descriptive string assigned to the Cal Set
<u>DwellTime</u>	ICalSet3	Returns the dwell time for the channel.
<u>FrequencyOffsetCWOverride</u>	ICalSet3	Reads state of CW Override (ON or OFF)
<u>FrequencyOffsetDivisor</u>	ICalSet3	Reads Frequency Offset Divisor value
<u>FrequencyOffsetFrequency</u>	ICalSet3	Reads Offset Frequency
<u>FrequencyOffsetMultiplier</u>	ICalSet3	Reads Frequency Offset Multiplier value
<u>FrequencyOffsetState</u>	ICalSet3	Reads Frequency Offset state (ON or OFF)
<u>IFBandwidth</u>	ICalSet3	Reads IF Bandwidth of the channel
<u>Item</u>	ICalSet6	Add or change a name-value pair in the Cal Set, or read the value associated with a name.
<u>LastModified</u>	ICalSet3	Reads the time stamp of when the file was last modified
<u>Name</u>	ICalSet4	Sets and returns the Cal Set name.
<u>NumberOfPoints</u>	ICalSet3	Returns the Number of Points of the channel.
<u>OutputPorts</u>	ICalSet5	Returns the port mapping used for the Cal Set.
<u>PowerSlope</u>	ICalSet3	Returns the Power Slope value.
<u>ReceiverAttenuator</u>	ICalSet3	Returns the value of the specified receiver attenuator control.
<u>StartFrequency</u>	ICalSet3	Returns the start frequency of the channel.
<u>StartPower</u>	ICalSet3	Returns the start power of the PNA when sweep type is set to Power Sweep.
<u>StimulusValues</u>	ICalSet3	Returns x-axis values for stimulus or response frequencies
<u>StopFrequency</u>	ICalSet3	Returns the stop frequency of the channel.

StopPower	ICalSet3	Returns the stop power of the PNA when sweep type is set to Power Sweep.
SweepGenerationMode	ICalSet3	Returns the method being used to generate a sweep: analog or stepped.
SweepTime	ICalSet3	Returns the sweep time of the analyzer.
SweepType	ICalSet3	Returns the type of X-axis sweep that is performed on a channel.
TestPortPower	ICalSet3	Returns the RF power level for the channel.
TestSetType	ICalSet5	Returns the Test Set type used for the Cal Set.

ICalSet History

Interface	Introduced with PNA Rev:
ICalSet	2.0
ICalSet2	3.0
ICalSet3	3.2
ICalSet4	6.0
ICalSet5	6.2
ICalSet6	9.30
ICalSet7	9.40

ICalData Interface

Description

Use this interface as an alternative to the ICalSet Interface to avoid using variants when transmitting data to and from the Cal Set

[Learn about reading and writing Calibration data.](#)

Methods	Interface	Description
	See History	
getErrorTermComplex	ICalData2	Superseded with getErrorTermComplexByString
getErrorTermComplexByString	ICalData3	Returns typed error term data by specifying the string name of the error term.
getStandardComplex	ICalData2	Superseded with getStandardComplexByString
getStandardComplexByString	ICalData3	Returns typed standard acquisition data by specifying the string name of the standard.
putErrorTermComplex	ICalData2	Superseded with putErrorTermComplexByString
putErrorTermComplexByString	ICalData3	Writes typed error term data by specifying the string name of the error term.
putStandardComplex	ICalData2	Superseded with putStandardComplexByString
putStandardComplexByString	ICalData3	Writes typed standard acquisition data by specifying the string name of the standard.
Properties		Description

None

History

Interface	Introduced with PNA Rev:
ICalData2	2.0
ICalData3	3.1

The original ICalData Interface was introduced with PNA 1.0 on the [Calibrator](#) Object.

ICalData2	2.0
ICalData3	3.1

Last modified:

2-May-2011	Added Get/Put ErrorTermStimulus
1-Nov-2006	New start and stop freq commands added

CalSets Collection

Description

A collection object that provides a mechanism for iterating through all the Cal Sets in the analyzer. There is no ordering to the items in the collection. Therefore make no assumptions about the formatting of the collection.

For the Item and Remove methods, you can specify either the Cal Set string name, or the integer item of the Cal Set in the collection.

Accessing the CalSets collection

Get a handle to the CalSets collection through the CalManager object with the app [GetCalManager](#) Method.

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim calsts As CalSets
Set calsts = app.GetCalManager.CalSets
```

See Also:

- [CalSet Object](#)
- [Collections in the Analyzer](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods	Interface	Description
Exists	ICalSets2	Returns whether the specified Cal Set exists
Item	ICalSets	Returns a handle to a CalSet object in the collection.
Remove	ICalSets	Deletes the Cal Set residing at position index in the collection.

Properties	Description
Count	ICalSets Returns the number of Cal Sets in the collection.

CalSets History

Interface	Introduced with PNA Rev:
ICalSets	1.0
ICalSets2	9.33

Last Modified:

28-Feb-2011 Added Exists (9.33)
30-Oct-2007 added item and remove note.

CalStandard Object

Description

Contains all of the settings that are required to modify a calibration standard.

For more information, read [Specifying Calibration Standards and Kits for Agilent Vector Network Analyzers \(Application Note 1287-11\)](#)

Accessing the CalStandard object

Get a handle to a standard with the calkit.[GetCalStandard](#) Method.

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim std As ICalStandard
Set std = app.ActiveCalKit.GetCalStandard(1)
std.Delay = 0.00000003
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Reading and Writing Calibration data](#)
- [Example Programs](#)

Methods

None

Properties	Interface	Description
	See History	
C0	ICalStandard	Sets and Returns the C0 (C-zero) value (the first capacitance value) for the calibration standard, when the Type is set to "naOpen".
C1	ICalStandard	Sets and Returns the C1 value (the second capacitance value) for the calibration standard, when the Type is set to "naOpen".
C2	ICalStandard	Sets and Returns the C2 value (the third capacitance value) for the calibration standard, when the Type is set to "naOpen".

C3	ICalStandard	Sets and Returns the C3 value (the fourth capacitance value) for the calibration standard, when the Type is set to "naOpen".
Delay	ICalStandard	Sets and Returns the electrical delay value for the calibration standard.
L0	ICalStandard	Sets and Returns the L0 (L-zero) value (the first inductance value) for the calibration standard, when the Type is set to "naShort".
L1	ICalStandard	Sets and Returns the L1 value (the second inductance value) for the calibration standard, when the Type is set to "naShort"..
L2	ICalStandard	Sets and Returns the L2 value (the third inductance value) for the calibration standard, when the Type is set to "naShort"..
L3	ICalStandard	Sets and Returns the L3 value (the third inductance value) for the calibration standard, when the Type is set to "naShort"..
Label	ICalStandard	Sets and Returns the label for the calibration standard.
loss	ICalStandard	Sets and Returns the insertion loss for the calibration standard.
Maximum Frequency	ICalStandard	Sets and Returns the maximum frequency for the calibration standard.
Medium	ICalStandard	Sets and Returns the media type of the calibration standard.
Minimum Frequency	ICalStandard	Sets and Returns the mininum frequency for the calibration standard.
Type	ICalStandard	Sets and Returns the type of calibration standard. Selections are: naOpen, naShort, naLoad, naThru, naArbitraryImpedance and naSliding.
TZReal	ICalStandard2	Sets and Returns the TZReal value (the Real Terminal Impedance value) for the calibration standard, when the Type is set to "naArbitraryImpedance".
TZImag	ICalStandard2	Sets and Returns the TZImag value (the Imaginary Terminal Impedance value) for the calibration standard, when the Type is set to "naArbitraryImpedance".
Z0	ICalStandard	Sets and Returns the characteristic impedance for the calibration standard.

ICalStandard History

Interface	Introduced with PNA Rev:
------------------	-------------------------------------

CalStandard	1.0
CalStandard2	3.0

Capabilities Object

Description

These properties return capabilities of the remote PNA.

Accessing the Capabilities object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim cap As Capabilities
Set cap = app.Capabilities
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [ICapabilities History](#)
- [Example Programs](#)

Methods	Interface
	See History
GetPortNumber Method	ICapabilities4 Returns the port number for the specified string port name.

Properties	Description
AvailableMeasurementClasses	ICapabilities7 Returns the measurement classes on the PNA
CpuRevision	ICapabilities6 Returns the CPU speed of the PNA
DirectoryPath	ICapabilities10 Returns the directory location.
DspRevision	ICapabilities6 Returns the DSP Revision number
DspFpgaRevision	ICapabilities6 Returns the DSP FPGA Revision number
FirmwareMajorRevision	ICapabilities Returns integer portion of firmware revision number.

<u>FirmwareMinorRevision</u>	ICapabilities	Return decimal portion of firmware revision number.
<u>FirmwareSeries</u>	ICapabilities	Returns the Alpha portion of the firmware revision number.
<u> GPIBPortCount</u>	ICapabilities3	Returns the number of GPIB ports (1 or 2)
<u>IFBWList</u>	ICapabilities8	Returns a list of supported IFBW values.
<u>InternalTestsetPortCount</u>	ICapabilities	Returns the number of PNA test ports.
<u>IsFrequencyOffsetPresent</u>	ICapabilities	Returns the presence of Frequency Offset Option 080 (True or False).
<u>IsReceiverStepAttenuatorPresent</u>	ICapabilities	Returns the presence of receiver step attenuators (True or False).
<u>IsReferenceBypassSwitchPresent</u>	ICapabilities	Returns the presence of the reference switch (True or False).
<u>MaximumFrequency</u>	ICapabilities	Returns the maximum frequency of the PNA.
<u>MaximumNumberOfChannels</u>	ICapabilities2	Returns the maximum possible number of Channels
<u>MaximumNumberOfPoints</u>	ICapabilities	Returns the maximum possible number of data points.
<u>MaximumNumberOfTracesPerWindow</u>	ICapabilities2	Returns the maximum possible number of traces per window
<u>MaximumNumberOfWindows</u>	ICapabilities2	Returns the maximum possible number of windows
<u>MaximumReceiverStepAttenuator</u>	ICapabilities	Returns the maximum amount of receiver attenuation.
<u>MaximumSourceALCPower</u>	ICapabilities	Returns the maximum amount of source ALC power.
<u>MaximumSourceStepAttenuator</u>	ICapabilities	Returns the maximum amount of source attenuation.
<u>MeasurementClassProperties</u>	ICapabilities8	Returns a handle to the MeasurementClassProperties Object
<u>MinimumFrequency</u>	ICapabilities	Returns the minimum frequency of the PNA.

<u>MinimumNumberOfPoints</u>	ICapabilities	Returns the minimum possible number of data points.
<u>MinimumReceiverStepAttenuator</u>	ICapabilities	Returns the minimum amount of receiver attenuation.
<u>MinimumSourceALCPower</u>	ICapabilities	Returns the minimum amount of source ALC power.
<u>ModelNumber</u>	ICapabilities10	Returns the model number of the PNA.
<u>ReceiverCount</u>	ICapabilities	Returns the number of receivers in the PNA.
<u>ReceiverStepAttenuatorStepSize</u>	ICapabilities	Returns the step size of the attenuator.
<u>ReceiverTemperature Property</u>	ICapabilities9	Returns the temperature on the receiver board.
<u>ResBWList</u>	ICapabilities8	Returns a list of Res BW values that are supported by IM Spectrum.
<u>SerialNumber</u>	ICapabilities10	Returns the serial number of the PNA.
<u>SourceStepAttenuatorStepSize</u>	ICapabilities5	Returns a value indicating the step size of the source attenuator.
<u>SourceCount</u>	ICapabilities	Returns the number of sources.
<u>SourcePortCount</u>	ICapabilities4	Returns the number of source ports.
<u>SourcePortNames</u>	ICapabilities4	Returns the string names of source ports.
<u>SourceStepAttenuatorStepSize</u>	ICapabilities5	Returns a value indicating the step size of the source attenuator.

ICapabilities History

I Interface	Introduced with PNA Rev:
ICapabilities	3.5
ICapabilities2	5.23
ICapabilities3	6.0
ICapabilities4	7.20
ICapabilities5	8.04
ICapabilities6	9.10
ICapabilities7	9.33
ICapabilities8	9.40
ICapabilities9	9.50
ICapabilities10	10.0

Channel Object

See [SourcePowerCalData Interface](#) for putting and getting typed source power calibration data.

Description

The channel object is like the engine that produces data. Channel settings consist of stimulus values like frequency, power, IF bandwidth, and number of points.

Accessing the Channel object

You can get a handle to a channel in a number of ways. But first you have to make sure that the channel exists. When you first startup the analyzer, there is one S11 measurement on channel 1. Thus there is only one channel in existence. You can do the following:

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim chan As IChannel
Set chan = app.ActiveChannel
```

or

```
Set chan = app.Channels(2)
```

The first method returns the channel object that is driving the active measurement. If there is no measurement, there may not be a channel. Once a channel is created, it does not go away. So if there once was a measurement (hence a channel), the channel will still be available.

If there is no channel you can create one in a couple ways. You can do the following:

```
Pna.CreateMeasurement( ch1, "S11", port1, window2)
```

or

```
Pna.Channels.Add(2)
```

The latter will have no visible effect on the analyzer. It will simply create channel 2 if it does not already exist.

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)

- [Superseded commands](#)

(**Bold** Methods or Properties provide access to a child object)

Methods	Interface	Description
	See History	
Abort	IChannel	Aborts the current measurement sweep on the channel.
ApplySourcePowerCorrectionTo	IChannel11	Copies an existing Source Power Calibration to another channel.
AveragingRestart	IChannel	Clears and restarts averaging of the measurement data.
Continuous	IChannel	The channel continuously responds to trigger signals.
CopyToChannel	IChannel2	Sets up another channel as a copy of this objects channel.
GetConverter	IChannel17	Returns a handle to a Converter object.
GetErrorCorrection	IChannel8	Returns the channel error correction state.
GetNumberOfGroups	IChannel3	Returns the number of groups a channel has yet to acquire.
GetPortNumber	IChannel13	Returns the port number for the specified string port name.
GetRxLevelingConfiguration	IChannel17	Returns a handle to a RxLevelingConfiguration object
getSourcePowerCalData	IChannel	Superseded with Get SourcePowerCalDataEx
getSourcePowerCalDataEx	IChannel14	Returns requested source power calibration data, if it exists.
GetSupportedALCModes	IChannel10	Returns a list of supported ALC modes
GetXAxisValues	IChannel	Returns the channel's X-axis values into a dimensioned Variant array.
GetXAxisValues2	IChannel	Returns the channel's X-axis values into a dimensioned NON-Variant array.
Hold	IChannel	Puts the Channel in Hold - not sweeping.
Next_IFBandwidth	IChannel	A function that returns the Next higher IF Bandwidth value.

NumberOfGroups	IChannel	Sets the Number of trigger signals the channel will receive.
Preset	IChannel	Resets the channel to factory defined settings.
PreviousIFBandwidth	IChannel	Returns the previous IF Bandwidth value.
putSourcePowerCalData	IChannel	Superseded with Put SourcePowerCalDataEx Method
putSourcePowerCalDataEx	IChannel4	Inputs source power calibration data to this channel for a specific source port.
SelectCalSet	IChannel	Specifies the Cal Set to use for the Channel
Single	IChannel	Channel responds to one trigger signal from any source (internal, external, or manual). Then channel switches to Hold.

Properties	Interface	Description
ALCLevelingMode	IChannel10	Set or return the ALC leveling mode.
AlternateSweep	IChannel	Sets sweeps to either alternate or chopped.
Attenuator	IChannel	Sets or returns the value of the attenuator control for the specified port number.
AttenuatorMode	IChannel	Sets or returns the mode of operation of the attenuator control for the specified port number.
AuxiliaryTrigger	IChannel10	Provides access to Auxiliary Triggering
Averaging	IChannel	Turns trace averaging ON or OFF for all measurements on the channel.
AveragingCount	IChannel	Returns the number of sweeps that have been averaged into the measurements.
AveragingFactor	IChannel	Specifies the number of measurement sweeps to combine for an average.
AverageMode	IChannel16	Sets Point or Sweep averaging.
BalancedTopology	IChannel6	Provides access to the topology of a balanced DUT.
Calibrator	IChannel4	Provides access to Unguided calibration.
CalSet	IChannel7	Provides access to the contents of a Cal Set
centerFrequency	IChannel	Sets or returns the center frequency of the channel.
		Shared with the Segment Object

<u>channelNumber</u>	IChannel	Returns the Channel number. Shared with the Measurement Object
<u>Converter</u>	IChannel21	Provides access to a mixer/converter object.
<u>CorrectionMethods</u>	IChannel21	Provides access to channel correction properties.
<u>CoupleChannelParams</u>	IChannel5	Turns ON and OFF Time Domain Trace Coupling.
<u>CouplePorts</u>	IChannel	Turns ON and OFF port power coupling.
<u>CustomChannelConfiguration</u>	IChannel12	Provides access to custom application objects.
<u>CWFrequency</u>	IChannel	Set the Continuous Wave (CW) frequency.
<u>DefinedRoles</u>	IChannel22	Returns the roles for which sources can be used for the channel.
<u>DwellTime</u>	IChannel	Sets or returns the dwell time for the channel. Shared with the Segment Object
<u>ErrorCorrection</u>	IChannel17	Attempts to sets error correction ON or OFF for all of the measurements on the channel.
<u>ExternalTriggerDelay</u>	IChannel6	Sets or returns the external trigger delay value for the channel.
<u>FastCWPointCount</u>	IChannel16	Enables Fast CW sweep and sets the number of data points for the channel.
<u>Fixturing</u>	IChannel6	Provides access to Port Ext, Embedding, and De-embedding functions.
<u>FOM Collection</u>	IChannel9	Provides access to Frequency Offset Measurements
<u>FrequencyOffsetDivisor</u>	IChannel2	
<u>FrequencyOffsetFrequency</u>	IChannel2	
<u>FrequencyOffsetMultiplier</u>	IChannel2	Superseded with <u>FOM</u> and <u>FOMRange</u>
<u>FrequencyOffsetCWOverride</u>	IChannel2	
<u>FrequencyOffsetState</u>	IChannel2	
<u>FrequencySpan</u>	IChannel	Sets or returns the frequency span of the channel. Shared with the Segment Object.
<u>IFBandwidth</u>	IChannel	Sets or returns the IF Bandwidth of the channel. Shared with the Segment Object.

<u>IFConfiguration</u>	IChannel4	Provides access to the IF gain and source path settings for the H11 Option.
<u>IsContinuous</u>	IChannel3	Returns whether or not a channel is in <u>continuous</u> mode.
<u>IsHold</u>	IChannel3	Returns whether or not a channel is in <u>hold</u> mode.
<u>MeasurementClass</u>	IChannel15	Returns the measurement class name.
<u>NumberOfPoints</u>	IChannel	Sets or returns the Number of Points of the channel. Shared with the Segment Object.
<u>Parent</u>	IChannel	Returns a handle to the parent object of the channel.
<u>PathConfiguration</u>	IChannel10	Provides access to path configuration switches and setting.
<u>PathConfigurationManager</u>	IChannel10	Provides access to path configuration file management.
<u>PointSweepState</u>	IChannel16	Turns point sweep ON or OFF for all measurements on the channel.
<u>PowerSlope</u>	IChannel	Sets or returns the Power Slope value.
<u>PowerSlopeState</u>	IChannel18	Turns power slope ON or OFF
<u>PulseGenerator</u>	IChannel10	Provides access to pulse generator configuration.
<u>PulseGeneratorID</u>	IChannel23	Returns the ID for the specified Pulse Generator name.
<u>PulseGeneratorNames</u>	IChannel23	Returns a list of configured Pulse Generator names.
<u>PulseMeasControl</u>	IChannel20	Provides access to pulse measurement settings.
<u>R1InputPath</u>	IChannel2	Throws internal reference switch.
<u>ReceiverAttenuator</u>	IChannel	Sets or returns the value of the specified receiver attenuator control.
<u>ReduceIFBandwidth</u>	IChannel5	Sets or returns the state of the <u>Reduced IF Bandwidth at Low Frequencies</u> setting.
<u>RoleDevice</u>	IChannel22	Sets and returns the source to be used in the specified role.
<u>RXLevelingConfiguration</u>	IChannel21	Provides access to the ReceiverLeveling Object
<u>Segments</u>	IChannel	Provides access to the Collection for iterating through the sweep segments of a channel.
<u>SourcePortCount</u>	IChannel13	Returns the number of source ports.

<u>SourcePortMode</u>	IChannel9	Sets the state of the PNA sources. (AUTO ON OFF)
<u>SourcePortNames</u>	IChannel13	Returns the string names of source ports.
<u>SourcePowerCalPowerOffset</u>	IChannel4	Sets or returns a power level offset from the PNA test port power.
<u>SourcePowerCorrection</u>	IChannel	Turns source power correction ON or OFF for a specific source port.
<u>StartFrequency</u>	IChannel	Sets or returns the start frequency of the channel. Shared with the Segment Object
<u>StartPower</u>	IChannel	Sets the start power of the analyzer when sweep type is set to Power Sweep.
<u>StartPowerEx</u>	IChannel13	Sets and reads the power sweep start power value for a specific port.
<u>StopFrequency</u>	IChannel	Sets or returns the stop frequency of the channel. Shared with the Segment Object
<u>StopPower</u>	IChannel	Sets the Stop Power of the analyzer when sweep type is set to Power Sweep.
<u>StopPowerEx</u>	IChannel13	Sets and reads the power sweep stop power value for a specific port.
<u>SweepDelay</u>	IChannel19	Sets the time to wait just before acquisition begins for each sweep.
<u>SweepGenerationMode</u>	IChannel	Sets the method used to generate a sweep: continuous ramp (analog) or discrete steps (stepped).
<u>SweepSpeedMode</u>	IChannel14	Set or returns the sweep speed mode.
<u>SweepTime</u>	IChannel	Sets the Sweep time of the analyzer.
<u>SweepType</u>	IChannel	Sets the type of X-axis sweep that is performed on a channel.
<u>TestPortPower</u>	IChannel	Sets or returns the RF power level for the channel. Shared with the Segment Object
<u>TriggerMode</u>	IChannel	Determines the measurement that occurs when a trigger signal is sent to the channel.

UserRangeMax	IChannel	Superseded - Use meas. UserRangeMax Sets the stimulus stop value for the specified User Range.
UserRangeMin	IChannel	Superseded - Use meas. UserRangeMin Sets the stimulus start value for the specified User Range.
XAxisPointSpacing	IChannel	Sets X-Axis point spacing for the active channel.

IChannel History

Interface	Introduced with PNA Rev:
IChannel	1.0
IChannel2	3.0
IChannel3	4.0
IChannel4	4.0
IChannel5	4.2
IChannel6	5.0
IChannel7	5.2
IChannel8	6.0
IChannel9	7.0
IChannel10	7.2
IChannel11	7.5
IChannel12	8.0
IChannel13	8.2
IChannel14	8.33
IChannel15	8.33
IChannel16	8.35
IChannel17	8.55
IChannel19	9.20
IChannel20	9.20
IChannel21	9.30
IChannel22	9.42
IChannel23	9.50
IChannel24	10.0

ISourcePowerCalData Interface

Description

Contains methods for putting source power calibration data in and getting source power calibration data out of the analyzer using typed data. The methods in this interface transfer data more efficiently than methods that use variant data. However, this interfaces is only usable from VB6, C, & C++. All other programming languages must use the methods on the Channel Object.

Note: Interface **ISourcePowerCalData** is abbreviated as **ISPCD** in the following table.

Methods	Interface	Description
	See History	
getSourcePowerCalDataScalar	ISPCD	Superseded - use PutSourcePowerCalDataScalarEx Method
getSourcePowerCalDataScalarEx	ISPCD2	Returns requested source power calibration data, if it exists.
putSourcePowerCalDataScalar	ISPCD	Superseded - use PutSourcePowerCalDataEx Method
putSourcePowerCalDataScalarEx	ISPCD2	Inputs source power calibration data to a channel, for a specific source port.
Properties		Description

None

ISourcePowerCalData History

Interface	Introduced with PNA Rev:
ISourcePowerCalData	2.0
ISourcePowerCalData2	4.0

Channels Collection

Description

A collection object that provides a mechanism for iterating through the channels

Collections are, by definition, unordered lists of like objects. You cannot assume that Channels.Item(1) is always Channel 1.

Accessing the Channels collection

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim chans As Channels
Set chans = app.Channels
```

See Also:

- [Channel Object](#)
- [Collections in the Analyzer](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods	Interface	Description
	See History	
Add	IChannels	An alternate way to create a measurement.
Hold	IChannels	Places all channels in Hold trigger mode.
Item	IChannels	Use to get a handle to a channel in the collection.
Remove	IChannels3	Delete a channel by specifying the index in the collection.
RemoveChannelNumber	IChannels3	Delete a channel by specifying the channel number.
Resume	IChannels2	Resumes the trigger mode of all channels that was in effect before sending the channels.Hold method.

Properties		Description
Count	IChannels	Returns the number of channels in the analyzer.
Parent	IChannels	Returns a handle to the current Application.
UnusedChannelNumbers	IChannels2	Returns an array of channel numbers that are NOT in use.
UsedChannelNumbers	IChannels2	Returns an array of channel numbers that are in use.

IChannels History

Interface	Introduced with PNA Rev:
IChannels	1.0
IChannels2	
IChannels3	9.30

ComColors Object

Description

Provides access to the methods and properties used to modify the PNA Display and Print colors.

Accessing the ComColors object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835X.Application")
Set displayColors = app.Preferences.DisplayColors
'or
'Set printColors = app.Preferences.PrintColors
displayColors.ActiveLabels = 657930
```

See Also:

- [ComTraceColors Object](#)
- [Modify Display Colors Example](#)
- [About PNA Display Colors](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)

(**Bold** Methods or Properties provide access to a child object)

Methods	Interface	
	See History	
LoadTheme	IComColors	Load a color theme from a disc file.
ResetTheme	IComColors	Resets the current theme to the default PNA colors.
StoreTheme	IComColors	Saves the current color theme to a disc file.
Properties		Description
ActiveLabels	IComColors	Sets labels and grid frame colors in the active window.
ActiveBackground	IComColors2	Set and return the background color for the active window on the PNA display or hardcopy print.

Background	IComColors	Set and return the background color for the inactive windows on the PNA display or hardcopy print.
FailedTraces	IComColors	Set and return the limit line color of failed traces.
Grid	IComColors	Set and return the inner lines of all grid in all windows.
InactiveLabels	IComColors	Set and return the Inactive (not selected) Window Labels.
Trace	IComColors	Provides access to the ComTraceColors Object for setting colors for the first 8 traces

IComColors History

I Interface	Introduced with PNA Rev:
IComColors	9.0
IComColors2	9.2

Last Modified:

7-Aug-2009 MX New topic

ComTraceColors Object

Description

Provides access to the methods and properties used to modify the PNA Display and Print colors.

Both the Display and Print [ComColor objects](#) contain 8 Trace objects (1 to 8).

'1st Trace' is NOT always Trace1 (Tr1). For example, the first trace in a window might be Tr2 which is drawn with the "1st Trace" pen.

The first 8 traces are drawn with the defined pen colors. The next eight traces reuse the same colors, and so forth. For example, if all traces are numbered sequentially, the 9th and 17th traces are drawn using the same color as the 1st trace.

Accessing the ComTraceColors object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835X.Application")
Set displayColors = app.Preferences.DisplayColors
'or
'Set printColors = app.Preferences.PrintColors
dim Trace1

Set Trace1 = displayColors.Trace(1)
Trace1.DataAndLimits = RGB(1,251,1)
```

See Also:

- [Modify Display Colors Example](#)
- [About PNA Display Colors](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [IComColors History](#)
- [Example Programs](#)

Methods**Interface**[See History](#)

None

Properties**Description**

DataAndLimits	IComTraceColors	Set and return the color of Data and Limit Lines for nth trace in a window.
Markers	IComTraceColors	Set and return the color of data trace markers for nth trace in a window.
Memory	IComTraceColors	Set and return the memory trace color for nth trace in a window.
MemoryMarkers	IComTraceColors	Set and return the color of memory trace markers for nth trace in a window.

IComTraceColors History

I Interface	Introduced with PNA Rev:
--------------------	-------------------------------------

IComTraceColors	9.0
-----------------	-----

Converter Object

Note: The Converter Object replaces the IMixer Interface .

Description

Contains the methods and properties to setup a mixer for ALL PNA Mixer/Converter applications .

Accessing the Converter Interface

```
Dim app as AgilentPNA835x.Application  
  
Dim chan as Channel  
  
Set chan = app.ActiveChannel  
  
Dim converter as Converter  
  
Set converter =chan.Converter
```

Scratch vs Applied Mixer Properties

Each mixer configuration has two sets of properties:

1. **Scratch mixer** contains the properties that have been set, but NOT YET applied. Send the Apply Method to copy these properties to the Applied mixer.
2. **Applied mixer** contains the properties that makeup the current mixer configuration.

Power settings are immediately applied to both the Scratch and Applied mixer.

A successful Calculate also performs an Apply.

Note: Queries always return the Applied mixer properties. Therefore, first send Apply Method before querying new settings.

See Also

- PNA Automation Interfaces
- The PNA Object Model
- Mixer Setup UI topic

Methods	Interface	Description
	See History	
AddSegment	Converter5	Add segments to the segment table (FCA Only)
Apply	Converter	Applies mixer settings.
AssignSourceToRole	Converter	Assigns a configured source to the specified role.
Calculate	Converter	Automatically calculate Input and Output frequencies for mixer setup.
DeleteAllSegments	Converter5	Remove all segments from the segment table. (FCA Only)
DeleteSegment	Converter5	Remove segments from the segment table. (FCA Only)
DiscardChanges	Converter	Cancels changes that have been made to the Converter setup.
GetSourceByRole	Converter	Returns the name of a source that is assigned to the specified role.
GetSourceRoles	Converter	Returns the defined role names ("RF2", "LO1").
LoadFile	Converter	Loads a previously-configured mixer attributes file (.mxr)
ReCalculate	Converter5	Repeats the last calculation that was performed.
SaveFile	Converter	Saves the settings for the mixer/converter test setup to a mixer attributes file.
SegmentCalculate	Converter5	Performs calculate on a specific segment. (FCA Only)

Properties	Description
ActiveXAxisRange	Converter Sets or returns the swept frequency range to display on the X-axis.

AvoidSpurs	Converter5	Sets and returns the state of the avoid spurs feature.
ConverterEmbeddedLO	Converter2	Provides access to the ConverterEmbeddedLO Object
DeviceInputPort	Converter6	Read the PNA port number which is connected to the DUT Input.
DeviceOutputPort	Converter6	Read the PNA port number which is connected to the DUT Output.
EnablePhase	Converter3	Sets and returns the state of SMC + Phase measurements and calibrations. (SMC Only)
IFDenominator	Converter4	Sets or returns the denominator value of the IF Fractional Multiplier.
IFNumerator	Converter4	Sets or returns the numerator value of the IF Fractional Multiplier.
IFSideband	Converter4	Select sum or difference for IF product.
IFStartFrequency	Converter4	Sets or returns the IF start frequency.
IFStopFrequency	Converter4	Sets or returns the IF stop frequency.
IncludeReverseSweep	Converter4	Sets whether to include SC12 sweeps during measurements.
InputDenominator	Converter	Sets or returns the denominator value of the Input Fractional Multiplier.
InputFixedFrequency	Converter	Sets or returns the mixer fixed Input frequency value.
InputNumerator	Converter	Sets or returns the numerator value of the Input Fractional Multiplier.
InputPower	Converter	Sets or returns the value of the Input Power.
InputRangeMode	Converter	Sets or returns the Input sweep mode.
InputStartFrequency	Converter	Sets or returns the start frequency of the mixer input.
InputStartPower	Converter4	Sets and returns the Start Power value of the mixer Input Power.

InputStopFrequency	Converter	Sets or returns the stop frequency of the mixer input.
InputStopPower	Converter4	Sets and returns the Stop Power value of the mixer Input Power.
IsInputGreaterThanLO	Converter	Specifies whether to use the Input frequency that is greater than the LO or less than the LO.
LODenominator	Converter	Sets or returns the denominator value of the LO Fractional Multiplier.
LOFixedFrequency	Converter	Sets or returns the fixed frequency of the specified LO.
LOName	Converter	Sets or returns the LO name.
LONumerator	Converter	Sets or returns the numerator value of the LO Fractional Multiplier.
LOPower	Converter	Sets or returns the value of the LO Power.
LORangeMode	Converter	Sets or returns the LO sweep mode to fixed or swept.
LOStage	Converter	Returns the number of stages.
LOStartFrequency	Converter	Sets or returns the start frequency of the specified LO.
LOStartPower	Converter	Sets or returns the start value of a LO Power sweep.
LOStopFrequency	Converter	Sets or returns the start frequency of the specified LO.
LOStopPower	Converter	Sets or returns the stop value of a LO Power sweep.
NominalIncidentPowerState	Converter3	Sets or returns whether to use nominal power or measure actual incident power. (SMC ONLY)
NormalizePoint	Converter3	Sets or returns the data point used for normalizing an SMC phase measurement. (SMC Only)
OutputFixedFrequency	Converter	Sets or returns the fixed frequency of the mixer output.

OutputRangeMode	Converter	Sets or returns the Output sweep mode.
OutputSideband	Converter	Sets or returns the value of the output sideband.
OutputStartFrequency	Converter	Sets or returns the start frequency of the mixer output.
OutputStopFrequency	Converter	Sets or returns the stop frequency of the mixer output.
SegmentCount	Converter5	Read the number of segments. (FCA Only)
SegmentFixedFrequency	Converter5	Set and read the CW Frequency for mixer segments in CW Sweep mode. (FCA Only)
SegmentFixedPower	Converter5	Set and return the fixed power level for all ranges for mixer segments. (FCA Only)
SegmentIFBandwidth	Converter5	Set and return the IF Bandwidth for the sweep segment. (FCA Only)
SegmentIsInputGreaterThanLO	Converter5	Set and return whether to use the Input frequency that is greater than the LO. (FCA Only)
SegmentMixingMode	Converter5	Set and return whether to set the mixing mode to high side or low side. (FCA Only)
SegmentPoints	Converter5	Sets and returns the number of data points to be measured in the sweep segment. (FCA Only)
SegmentRangeMode	Converter5	Sets or returns the sweep mode of the segment (all ranges). (FCA Only)
SegmentStartFrequency	Converter5	Set and return the Start frequency for mixer segments. (FCA Only)
SegmentState	Converter5	Set and return the ON/OFF state for mixer segments. (FCA Only)
SegmentStopFrequency	Converter5	Set and return the Stop frequency for mixer segments. (FCA Only)

Converter History

Interface	Introduced with PNA Rev:
Converter	8.55
Converter2	9.0
Converter3	9.2
Converter4	9.30
Converter5	9.33
Converter6	9.40

Last Modified:

- 3-Jan-2012 Removed superseded
- 24-May-2011 Added Converter 6
- 3-May-2011 Superseded
- 7-Jan-2011 FCA2 Update
- 2-Feb-2009 New topic

ConverterEmbeddedLO Object

Description

Provides access to the properties that allow IMDx and IMSpectrum measurements of converters that contain an embedded LO.

This interface contains all the same properties and methods of the Embedded LO interface (used for FCA measurements) EXCEPT access to the EmbeddedLODiagnostic Object .

Accessing the ConverterEmbeddedLO Interface

```
Access the Interface through the Converter Object.

Dim app

Set app = CreateObject("AgilentPNA835x.Application")

app.Reset

' Create a Measurement object, in this case using the IMeasurement interface

Dim meas

app.CreateCustomMeasurement 1, "Swept IMD Converters", -1

set meas = app.activemeasurement

dim converter

set converter = app.ActiveChannel.GetConverter

dim elo

set elo = converter.ConverterEmbeddedLO

elo.IsOn = 1
```

See Also:

[PNA Automation Interfaces](#)

[The PNA Object Model](#)

[Making Embedded LO Measurements](#)

Methods	Interface	Description
	ICConverterEmbeddedLO is abbreviated as ICELO	
	See History	

ResetLOFrequency	ICELO	Reset LO Delta frequency.
ResetTuningParameters	ICELO	Resets the tuning parameters to their defaults.

Properties		Description
BroadbandTuningSpan	ICELO	Set broadband sweep span.
IsOn	ICELO	Set and return Embedded LO ON OFF.
LOFrequencyDelta	ICELO	Sets and returns LO delta frequency.
MaxPreciseTuningIterations	ICELO	Sets and returns precise tuning iterations.
NormalizePoint	ICELO	Sets and returns tuning point.
PreciseTuningTolerance	ICELO	Sets and returns precise tuning tolerance.
TuningIFBW	ICELO	Sets and returns the IF Bandwidth for tuning sweeps.
TuningMode	ICELO	Sets and returns the method used to determine the embedded LO Frequency.
TuningSweepInterval	ICELO	Set how often a tuning sweep is performed.

ICELO History

Interface	Introduced with PNA Rev:
ICELO	9.00

Last Modified:

12-Aug-2009 MX New topic

CorrectionMethods Object

Description

These methods and properties control various error-correction settings for a channel.

Accessing the object

```
Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel
Dim corrMethods as CorrectionMethods
corrMethods = chan.CorrectionMehods
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods	Interface	Description
	See History (below)	

None

Properties	Description
------------	-------------

[MatchCorrectPower](#) ICorrectionMethods Turns match-correction ON or OFF after performing an Enhanced Power Cal.

IAuxTrigger History

Interface	Introduced with PNA Rev:
ICorrectioMethods	9.30

Last Modified:

12-Mar-2014 MX New topic

DCStimulus Object

Description

The DCStimulus object allows you to make DC Source settings for each channel.

Accessing the DCStimulus Object

You can obtain a handle to a DCStimulus object through the Channel object.

```
dim app
Set app = CreateObject("AgilentPNA835x.Application")
dim chan
Set chan = app.ActiveChannel
chan.NumberofPoints = 3
dim DC
Set DC = chan.DCStimulus
DC.EnableAllOutput = True
DC.State("A01")= True
```

See Also

[Learn about DC Source Control](#)

[Configure an External DC Device](#)

[The PNA Object Model](#)

Methods**Description**

None

Properties**Description**

[EnableAllOutput](#)

Sets and returns the ON / OFF state of all DC sources for the channel.

[ListData](#)

Sets and returns the stimulus value array of the DC source for the channel.

[Sources](#)

Returns the names of the configured DC sources for the channel.

[Start](#)

Sets and returns start DC value for the specified DC source.

[State](#)

Sets and returns the ON / Off / Port state of the specified DC source.

[Stop](#)

Sets and returns Stop DC value for the specified DC source.

Target

For future use

DCStimulus History**Interface Introduced with PNA Rev:**

IDCStimulus 9.5

Last modified:

20-Feb-2012 New topic

E5091Testsets Collection

Description

Two testsets can be connected and controlled by the PNA at any time.

The item number in the testsets collection is set by the DIP switches on the testset rear-panel. The valid item numbers are 1 and 2. If the testset DIP switches are set to 1, then item number in the collection is 1, and so forth. See your E5091A documentation for more information.

If the specified testset is not connected to USB or not ON, then setting [Enabled](#) = True will return an error. All other properties can be set when the testset is not connected.

Accessing the E5091Testsets collection

Child of the **Application** Object. Get a handle to one of the [E5091Testset objects](#) by specifying an item of the collection.

```
Dim pna
Set pna = CreateObject("AgilentPNA835x.Application")
Dim testsets As E5091Testsets
Set testsets = pna.E5091Testsets
Dim tset1 As E5091Testset
Set tset1 = testsets(1)
```

See Also:

- [E5091Testset Control COM Example](#)
- [E5091Testset Object](#)
- [Collections in the Analyzer](#)
- [The PNA Object Model](#)

Methods	Description
Item	Use to get a handle to a testset in the collection.

Properties	Description
Count	Returns the number of items in a collection of objects.
Parent	Returns a handle to the current naNetworkAnalyzer application.

E5091Testsets History

Interface	Introduced with PNA Rev:
IE5091Testsets	5.2

E5091Testset Object

Description

There can be two test sets connected and controlled by the PNA at any time.

The item number in the testsets collection is set by the DIP switches on the test set rear-panel. The valid item numbers are 1 and 2. If the test set DIP switches are set to 1, then item number in the collection is 1, and so forth. See your E5091A documentation for more information.

If the specified test set is not connected to USB or not ON, then setting [Enabled](#) = True will return an error. All other properties can be set when the test set is not connected.

Accessing the E5091Testset object

Child of the **Application** Object. Get a handle to a E5091Testset object by specifying an item of the collection.

```
Dim pna
Set pna = CreateObject("AgilentPNA835x.Application")
Dim testsets As E5091Testsets
Set testsets = pna.E5091Testsets
Dim tset1 As E5091Testset
Set tset1 = testsets(1)
```

See Also:

- [E5091Testset Control COM Example](#)
- [E5091 TestSet Control](#)
- [E5091Testsets Collection](#)
- [TestsetControl Object](#) (for different test sets)
- [The PNA Object Model](#)

Methods	Description
---------	-------------

None

Properties	Description
------------	-------------

ControlLines	Sets the control lines of the specified E5091A.
Enabled	Enables and disables (ON/OFF) the port mapping and control line output of the specified testset.
ID	Returns the test set ID number.
NumberOfPorts	Reads the number of ports (7 or 9) that are on the specified E5091A test set.
OutputPort	Switches an input to one of the valid outputs on the specified E5091A.
ShowProperties	Turns ON and OFF the display of the test set control status bar.

E5091Testset History

Interface	Introduced with PNA Rev:
IE5091Testset	5.2

ECalModule Object

Allows access to ECal modules that are connected to the PNA.

Accessing the ECalModule object

Get a handle to a ECalModule object by using the ECalModules collection. This is done through the CalManager object with the app.[GetCalManager](#) Method.

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim pna
pna.Preset

Const chanNum = 1

pna.Channels(chanNum).StopFrequency = 20E9 ' for a 20 GHz ECal mod

Const pnaPortNumber = 1

Const ecalCharacterizationNum = 0

Dim calMgr
Set calMgr = pna.GetCalManager

Dim ecalPortNumber ' The returned ECal port number is a 1-based number
' (1 = Port A, 2 = Port B, etc)

ecalPortNumber = calMgr.ECalModules(1).AutoOrient(chanNum, pnaPortNumber,
ecalCharacterizationNum)

MsgBox "ECal port number attached to PNA port 1 = " & ecalPortNumber
```

See Also:

- [ECalModules Collection](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods	Interface	Description
	See History	

AutoOrient	IECalModule	Returns the orientation (which ECal port is connected to which PNA port) outside of the context of a calibration.
----------------------------	-------------	---

Properties	Description
None	

IECalModule History

Interface	Introduced with PNA Rev:
IECalModule	8.50

Last Modified:

5-Mar-2009 MX New topic

ECalModules Collection

Description

A collection that provides access to ECal modules that are connected to the PNA.

Accessing the ECalModules collection

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)
Dim eCalMods As ECalModules
Set eCalMods = app.GetCalManager.ECalModules
```

See Also:

- [Using ECal](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods

[Item](#) Use to get a handle to a [ECalModule Object](#) in the collection.

Properties

Properties	Description
------------	-------------

[Count](#) Returns the number of objects in the collection.

[Parent](#) Returns a handle to the [CalManager](#) Object

IECalModules History

Interface	Introduced with PNA Rev:
IECalModules	8.50

Last Modified:

7-Apr-2009 MX New topic

ECalUserCharacterizer Object

Description

Controls the settings used to perform an ECal User Characterization. An S-Parameter channel must already be calibrated. These commands will then measure the ECal module with adapters, cables, or fixtures to be included in the User Characterization, allow descriptive text to be entered, then save the User Characterization to the ECal module.

Up to 12 User Characterizations can be stored in an ECal module.

You can NOT perform a **remote** User Characterization of a 4-port ECal module using a 2-port PNA. This can only be done from the front panel user interface.

Accessing the ECalUserCharacterizer Interface

Access the Interface through the ICalManager Object.

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application")

Dim mgr as ICalManager
Set mgr = app.GetCalManager
Dim ecalCharacterizer
Set ecalCharacterizer = mgr.GetECalUserCharacterizer()
```

See Also:

Example: Perform an ECal User Characterization

[PNA Automation Interfaces](#)

[The PNA Object Model](#)

[About User Characterization](#)

Methods	Interface	Description
	See History	
AcquireStep	ECalUserCharacterizer	Measure the ECal module.
GenerateSteps	ECalUserCharacterizer	Returns the number of steps required to complete the calibration.

GetStepDescription	ECalUserCharacterizer	Returns the description of the specified step in the calibration process.
Initialize	ECalUserCharacterizer	Superseded with InitializeEx Method
InitializeEx	ECalUserCharacterizer2	Initiates a User Characterization of an ECal module.
SaveToDiskMemory	ECalUserCharacterizer2	Saves the User Characterization to PNA disk memory.
SaveToECal	ECalUserCharacterizer	Saves the User Characterization to the ECal module.

Properties		Description
CharacterizationNumber	ECalUserCharacterizer	Sets and reads the number to which the user characterization will be stored in the ECal module.
ConnectorType	ECalUserCharacterizer	Sets or queries the connector type for the specified port.
ECalID	ECalUserCharacterizer	Select the model and serial number of the ECal module to be characterized.
InSituCharacterization	ECalUserCharacterizer3	Sets or returns whether the CalPod module will be characterized as an in situ device.
PortDescription	ECalUserCharacterizer	Sets and reads the description of the adapters, cable, or fixture to be included in the user characterization.
SupportsInSituCharacterization	ECalUserCharacterizer3	Returns whether the device is a CalPod module

UserDescriptionofPNA	ECalUserCharacterizer	Sets and reads a user description of the PNA used to perform the User Characterization.
UserName	ECalUserCharacterizer	Sets and reads the description of the person and/or company who is producing the ECal user characterization.
ValidConnectorTypes	ECalUserCharacterizer	Returns a list of connector names that are valid for use with user-characterized ECal modules.

IECalUserCharacterizer History

Interface	Introduced with PNA Rev:
IECalUserCharacterizer	8.33
IECalUserCharacterizer2	9.00

Last Modified:

2-Nov-2008 MX New topic

EmbeddedLO Object

Description

Provides access to the properties that allow measurement of mixers that contain an embedded LO.

Accessing the EmbeddedLO Interface

Access the Interface through the IMixer Object.

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application")
app.Preset

' FCA Measurements can't share the channel with standard measurements
' Because preset creates a single measurement in channel 1, we first delete the
standard measurement
Dim standardMeas As IMeasurement
Set standardMeas = app.ActiveMeasurement
standardMeas.Delete

' Create a Measurement object, in this case using the IMeasurement interface
Dim meas As IMeasurement
Set meas = app.CreateCustomMeasurementEx(1, "SMC_Forward.SMC_ForwardMeas", "SC21")

' See if this measurement object supports IMixer
Dim mixer As IMixer
Dim embeddedLO
Set embeddedLO = mixer.EmbeddedLO
```

Note: The **Find Now** feature on the Embedded LO dialog is performed remotely by doing a 'Single Sweep' with ELO enabled. See example program .

See an example program that shows how to create and calibrate a standard SMC or VMC measurement or a fixed output SMC measurement.

See Also:

[Example program](#)

[PNA Automation Interfaces](#)

[The PNA Object Model](#)

[Making Embedded LO Measurements](#)

Methods	Interface	Description
		See History

ResetLOFrequency	IEmbeddedLO	Reset LO Delta frequency.
ResetTuningParameters	IEmbeddedLO	Resets the tuning parameters to their defaults.

Properties	Description
------------	-------------

BroadbandTuningSpan	IEmbeddedLO	Set broadband sweep span.
EmbeddedLODiagnostic	IEmbeddedLO	Provides access to the status of tuning sweeps.
IsOn	IEmbeddedLO	Set and return Embedded LO ON OFF.
LOFrequencyDelta	IEmbeddedLO	Sets and returns LO delta frequency.
MaxPreciseTuningIterations	IEmbeddedLO	Sets and returns precise tuning iterations.
NormalizePoint	IEmbeddedLO	Sets and returns tuning point.
PreciseTuningTolerance	IEmbeddedLO	Sets and returns precise tuning tolerance.
TuningIFBW	IEmbeddedLO	Sets and returns the IF Bandwidth for tuning sweeps.
TuningMode	IEmbeddedLO	Sets and returns the method used to determine the embedded LO Frequency.
TuningSweepInterval	IEmbeddedLO	Set how often a tuning sweep is performed.

IEmbeddedLO History

Interface	Introduced with PNA Rev:
IEmbeddedLO	7.21

EmbeddedLODiagnostic Object

Description

Allows access to the properties that provide information about the broadband and precise tuning of an embedded LO.

Accessing the EmbeddedLODiagnostic Interface

Access the Interface through the EmbeddedLO Object.

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application")
app.Preset

' FCA Measurements can't share the channel with standard measurements
' Because preset creates a single measurement in channel 1, we first delete the
standard measurement
Dim standardMeas As IMeasurement
Set standardMeas = app.ActiveMeasurement
standardMeas.Delete

' Create a Measurement object, in this case using the IMeasurement interface
Dim meas As IMeasurement
Set meas = app.CreateCustomMeasurementEx(1, "SMC_Forward.SMC_ForwardMeas", "SC21")

' See if this measurement object supports IMixer
Dim mixer As IMixer
Dim embeddedLO
Set embeddedLO = mixer.EmbeddedLO

Dim embeddedLODiagnostic
Set embeddedLODiagnostic = embeddedLO.EmbeddedLODiagnostic
```

See an example program that shows how to create and calibrate a standard SMC or VMC measurement or a fixed output SMC measurement.

See Also:

[PNA Automation Interfaces](#)

[The PNA Object Model](#)

[Making Embedded LO Measurements](#)

[EmbeddedLO Object](#)

Methods	Interface	Description
	See History	

Clear IELODiag Clear current diagnostic information.

Properties	Description
------------	-------------

IsMarkerOn IELODiag Was a marker was used for a tuning sweep?

LODeltaFound IELODiag Returns the LO frequency delta from this tuning sweep.

NumberOfSweeps IELODiag Get number of tuning sweeps.

MarkerAnnotation IELODiag Get the marker annotation.

MarkerPosition IELODiag Get the marker X-axis position.

Parameter IELODiag Returns the tuning sweep parameter name.

StatusAsString IELODiag Get result of the last tuning sweeps.

StepData IELODiag Get a tuning sweep data.

StepTitle IELODiag Returns the tuning sweep title.

XAxisAnnotation IELODiag Get the tuning sweep X axis annotation.

XAxisStart IELODiag Get the Start sweep value.

XAxisStop IELODiag Get the Stop sweep value.

YAxisAnnotation IELODiag Get the tuning sweep Y axis annotation.

History

Interface	Introduced with PNA Rev:
IEmbeddedLODiagnostic	7.21

ENRFile Object

Description

Provide commands for creating or editing an ENR file. This is rarely necessary as ENR files, which contain factory calibrated data, are typically provided by the manufacturer of the noise source.

[Learn more about Noise Figure Application](#)

Accessing the ENRFile object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim enr As ENRFile
Set enr = app.ENRFile
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Program](#)

Methods	Interface	Description
	See History	
GetENRData	IENRFile	Read the ENR calibration data from PNA memory.
PutENRData	IENRFile	Write the ENR calibration data to PNA memory.
LoadENRFile	IENRFile	Recalls an ENR file from disk into PNA Memory.
SaveENRFile	IENRFile	Saves an ENR file from PNA memory to disk.

Properties	Interface	Description
	See History	
ENRID	IENRFile	Sets and returns ID of ENR table.
ENRSN	IENRFile	Sets and returns the serial number of the noise source.

IENRFile History

Interface	Introduced with PNA
IENRFile	Rev: 8.0

Last Modified:

2-Aug-2007 MX New topic

Equation Object

Description

Provide commands for creating an equation.

[Learn more about Equation Editor](#)

Accessing the Equation object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim eq As Equation
Set eq = app.Equation
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)

Methods	Interface	Description
	See History	
GetLibraryFunctions	IEquation2	Returns the functions in an imported (loaded) DLL.
ImportLibrary	IEquation2	Imports an Equation Editor DLL.
IsLibraryImported	IEquation2	Returns whether a DLL has been imported into the PNA.
RemoveLibrary	IEquation2	Removes an imported an Equation Editor DLL from the PNA.

Properties	Interface	Description
	See History	
Text	IEquation	Sets the Equation
State	IEquation	Sets the Equation enabled state
Valid	IEquation	Returns whether the equation is presently valid.

Example Program using these commands:

```
Dim na
Dim meas
Set na = CreateObject("AgilentPNA835x.Application")
Set meas = na.ActiveMeasurement
'Define the measurement
meas.Equation.Text = "mysillyequ=sqrt(AR1_1)"
'Check to see if the equation is valid
valid_e = meas.Equation.Valid
MsgBox valid_e
'Turn on the Equation Editor
meas.Equation.State = True
```

IEquation History

Interface	Introduced with PNA Rev:
IEquation	6.03

Last Modified:

4-Dec-2007 Added example

ExternalDCDevice Object

Description

The ExternalDCDevice object allows you to set unique properties for a DC Meter or DC Source.

Accessing the ExternalDCDevice Object

You can obtain a handle to a ExternalDCDevice Object through [ExtendedProperties](#)

```
dim app

Set app = CreateObject("AgilentPNA835x.Application")

dim externalDevices

Set externalDevices = app.ExternalDevices

dim devicecount

devicecount = externalDevices.count

'***** Configure a DC Meter *****

externalDevices.Add "MyDCMeter"

dim newExtDCMeter

Set newExtDCMeter = externalDevices.Item("MyDCMeter")

newExtDCMeter.DeviceType = "DC Meter"

newExtDCMeter.Driver = "DCMeter"

'newExtDCMeter.Active = True

newExtDCMeter.IOConfiguration= "GPIB0::16::INSTR"

dim extDCMtr

Set extDCMtr = newExtDCMeter.ExtendedProperties

extDCMtr.PointDwell = .05

'***** Configure a DC Supply *****

externalDevices.Add "MyDCSupply"

dim newExtDCSupply
```

```

Set newExtDCSupply = externalDevices.Item("MyDCSupply")

newExtDCSupply.DeviceType = "DC Source"

newExtDCSupply.Driver = "DCSource"

'newExtDCSupply.Active = True

newExtDCSupply.IOConfiguration= "GPIB0::16::INSTR"

dim extDCSupply

Set extDCSupply = newExtDCSupply.ExtendedProperties

extDCSupply.PointDwell = .05

```

See Also:

[Configure an External Device](#)

[The PNA Object Model](#)

Methods	Description
---------	-------------

None

Properties	Description
------------	-------------

- [DCCorrection](#) Sets and returns correction ON/OFF.
- [DCOffset](#) Sets and returns the offset correction value.
- [DCScale](#) Sets and returns the scale correction value.
- [DCType](#) Sets and returns the DC Type (Units).
- [PointDwell](#) Sets and returns the "Dwell Before/After Point" value.
- [SweepDwell](#) Sets and returns the "Dwell Before Sweep" value.

ExternalDCDevice History

Interface	Introduced with PNA Rev:
IExternalDCDevice	9.5

Last modified:

12-Mar-2014 New topic

ExternalDevice Object

Description

ExternalDevice objects allow you to set properties and methods for each external device.

Accessing the ExternalDevice Object

Obtain a handle to an External Device by specifying an item in the [External Devices collection](#).

Although the following example shows how to create a PMAR external device, you can substitute any supported external device for "PMAR" and DeviceType: "Power Meter".

Supported External Device Objects

[ExternalSource Object](#)

[ExternalDCDevice Object](#)

[PowerSensorAsReceiver Object](#)

[ExternalPulseGenerator Object](#)

```
dim app

Set app = CreateObject("AgilentPNA835x.Application")

dim externalDevices

Set externalDevices = app.ExternalDevices

dim devicecount

devicecount = externalDevices.count

externalDevices.Add "NewPMAR"

dim newExternalDevice

Set newExternalDevice = externalDevices.Item("NewPMAR")

newExternalDevice.DeviceType = "Power Meter"

newExternalDevice.IOConfiguration= "GPIB0::14::INSTR"

newExternalDevice.IOEnable = true
```

See Also:

[The PNA Object Model](#)

Learn how to [Configure an External Device](#)

Methods	Interface	Description
	See History	
LoadFile	IExternalDevice2	Recalls an external device configuration file from the PNA hard drive. Currently, only DC Meter and DC Supply devices are supported.
SaveFile	IExternalDevice2	Saves an external device configuration file to the PNA hard drive. Currently, only DC Meter and DC Supply devices are supported.

Properties		Description
Active	IExternalDevice	Sets and returns the state of activation for an external device.
DeviceType	IExternalDevice	Sets and returns the DeviceType (source, power meter) for the external device.
Driver	IExternalDevice	Sets and returns the external device driver (model).
ExtendedProperties	IExternalDevice	Provides access to properties that are unique to the external device type.
IOConfiguration	IExternalDevice	Sets and returns the method of communication and address for the external device.
IOEnable	IExternalDevice	Sets and returns whether an external device is available for IO communication with the PNA.
Name	IExternalDevice	Sets and returns the name of the External Device.
TimeOut	IExternalDevice	Sets and returns the time out value for communication with the external device.

ExternalDevice History

Interface	Introduced with PNA Rev:
------------------	---------------------------------

IExternalDevice	9.0
IExternalDevice2	9.50

Last Modified:

14-Mar-2012	Added links to other device types
31-Jul-2009	MX New topic (9.0)

ExternalDevices Collection

Description

ExternalDevices collection provides access to an ExternalDevice object.

Accessing the ExternalDevices collection

The ExternalDevices collection is a property of the main **Application** Object. You can obtain a handle to an External Device by specifying an item in the collection.

```
dim app

Set app = CreateObject("AgilentPNA835x.Application")

dim externalDevices

Set externalDevices = app.ExternalDevices

dim devicecount

devicecount = externalDevices.count

externalDevices.Add "NewPMAR"

dim newExternalDevice

Set newExternalDevice = externalDevices.Item("NewPMAR")

newExternalDevice.DeviceType = "Power Meter"

newExternalDevice.IOConfiguration= "GPIB0::14::INSTR"

newExternalDevice.IOEnable = true
```

See Also:

Example: [Create a PMAR Device and Measurement](#)

[Configure an External Device](#)

[ExternalDevice Object](#)

[The PNA Object Model](#)

Methods	Interface	Description
	See History	

Add	IExternalDevices	Adds an external device to the system.
Item	IExternalDevices	Use to get a handle to an external device in the collection.
Remove	IExternalDevices	Removes an external device from the system

Properties	Description
------------	-------------

Count	IExternalDevices	Returns the number of devices in the ExternalDevices collection.
DeviceNames	IExternalDevices2	Returns the device names in the collection
HasItem	IExternalDevices	Returns a value indicating whether the specified external devices is configured.
IsDevicePresent	IExternalDevices2	Returns whether the named device is present on the bus for which it is configured.
Items	IExternalDevices	Returns an array of configured devices in the system.
Parent	IExternalDevices	Returns a handle to the current application object .

IExternalDevices History

Interface	Introduced with PNA Rev:
IExternalDevices	9.0
IExternalDevices2	9.50

Last Modified:

30-Jul-2009 MX New topic

ExternalPulseGenerator Object

Description

The ExternalPulseGenerator object allows you to set unique properties and methods for each external pulse generator.

Accessing the ExternalPulseGenerator Object

You can obtain a handle to an ExternalSource Object through [ExtendedProperties](#)

```
dim app

Set app = CreateObject("AgilentPNA835x.Application")

dim externalDevices

Set externalDevices = app.ExternalDevices

dim devicecount

devicecount = externalDevices.count

externalDevices.Add "81110"

dim newExternalDevice

Set newExternalDevice = externalDevices.Item("81110")

newExternalDevice.DeviceType = "Pulse Generator"

newExternalDevice.Driver = "AGPULSEGEN"

newExternalDevice.Active = True

newExternalDevice.IOConfiguration= "GPIB0::19::INSTR"

dim extPulseGen

Set extPulseGen = newExternalDevice.ExtendedProperties

extPulseGen.HighAmplitude = 3
```

See Also:

[Configure an External Device](#)

[The PNA Object Model](#)

Methods	Interface	Description
---------	-----------	-------------

None

Properties	Description
------------	-------------

HighAmplitude	ExternalPulseGenerator	Sets the High amplitude (voltage).
LoadImpedance	ExternalPulseGenerator	Sets the Load impedance.
LowAmplitude	ExternalPulseGenerator	Sets the Low amplitude (voltage).
MasterMode	ExternalPulseGenerator2	Sets the master mode for the ext. pulse generator.
OutputChannel	ExternalPulseGenerator	Sets the Output channel (port) of the pulse generator.
SourceImpedance	ExternalPulseGenerator	Sets the Source impedance.

ExternalPulseGenerator History

Interface	Introduced with PNA Rev:
IExternalPulseGenerator	9.50
ExternalPulseGenerator2	9.50

ExternalSource Object

Description

The ExternalSource object allows you to set unique properties and methods for each external source.

Accessing the ExternalSource Object

You can obtain a handle to an ExternalSource Object through [ExtendedProperties](#)

```
dim app
Set app = CreateObject("AgilentPNA835x.Application")
dim externalDevices
Set externalDevices = app.ExternalDevices
dim devicecount
devicecount = externalDevices.count
externalDevices.Add "NewPSG"
dim newExternalDevice
Set newExternalDevice = externalDevices.Item("NewPSG")
newExternalDevice.DeviceType = "Source"
newExternalDevice.IOConfiguration= "GPIB0::14::INSTR"
dim PSG
Set PSG = newExternalDevice.ExtendedProperties
PSG.DwellPerPoint = 5
```

See Also:

[Configure an External Device](#)

[The PNA Object Model](#)

Methods**Description**

None

Properties**Description**

[DwellPerPoint](#)

Sets and returns the dwell time for an external source.

[TriggerMode](#)

Sets and returns the trigger mode (Software / Hardware) for an external source.

[TriggerPort](#)

Sets and returns the PNA port through which an external source is to be triggered.

ExtendedProperties History**Interface Introduced with PNA Rev:**

IExternalSource 9.0

Last Modified:

31-Jul-2009 MX New topic

ExternalTestsets Collection

Description

ExternalTestsets collection provides access to a TestsetControl object. Only one external testset can be controlled by the PNA at any time.

Accessing the ExternalTestsets collection

The ExternalTestsets collection is a property of the main **Application** Object. You can obtain a handle to a testset by specifying an item in the collection.

Visual Basic Example

```
Dim pna
Dim testsets As ExternalTestsets
Dim tset1 As TestsetControl
Set pna = CreateObject("AgilentPNA835x.Application")
Set testsets = pna.ExternalTestsets
Set tset1 = testsets(1)
' make COM calls on tset1 object
End Sub
```

See Also:

[ExternalTestset Control COM Example](#)

[About External TestSet Control](#)

[TestsetControl Object](#)

[The PNA Object Model](#)

Methods	Description
Add	Adds a testset to the collection and loads a test set configuration file.
Item	Use to get a handle to a testset in the collection.
TestsetCatalog	Returns a list of supported test sets.

Properties	Description
Count	Returns the number of items in a collection of objects.
Parent	Returns a handle to the current naNetworkAnalyzer application.

ExternalTestsets History

Interface	Introduced with PNA Rev:
IExternalTestsets	6.0
IExternalTestsets	6.2

FIFO Object

Description

These properties control the First IN, First OUT (FIFO) buffer settings for the PNA-X and N5264A.

The 4 GB FIFO data buffer is available with Option 118 on the [PNA-X](#) and [N5264A](#).

Accessing the FIFO object

```
Dim app as AgilentPNA835x.Application
Dim fifo as FIFO
Set fifo = app.FIFO
```

See Also:

- [About FIFO](#)
- [FIFO example program](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)

Methods		Description
Clear	IFIFO	Clears the FIFO buffer

Properties	Interface	Description
	See History	
Data Property	IFIFO	Reads the next specified number of data points from the FIFO buffer.
DataCount Property	IFIFO	Returns the total number of data points in the FIFO buffer.
DataInCompactForm	IFIFO	Reads data from the FIFO buffer. Same as Data but in a compact form.
State	IFIFO	Turns FIFO ON and OFF

IFIFO History

Interface

**Introduced with PNA
Rev:**

IFIFO

8.35

Last Modified:

1-May-2013 Modified accessing example

6-Oct-2008 MX New topic

Fixturing Object

Description

Contains the properties for Embedding and De-embedding test fixtures.

Accessing the Fixturing object

```
Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel
Dim fixt as Fixturing
Set fixt = chan.Fixturing
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [About Fixturing](#)
- [Example Programs](#)

Methods		Description
AutoPortExtMeasure	IFixturing2	Measures either an OPEN or SHORT standard.
AutoPortExtReset	IFixturing2	Clears old port extension delay and loss data.
NetworkPortMap	IFixturing2	Set the port mapping for a 4-port SNP file to be embedded.

Properties	Interface	Description
	See History	
AutoPortExtConfig	IFixturing2	Sets the frequency span that is used to calculate Automatic Port Extension.
AutoPortExtDCOffset	IFixturing2	Specifies whether or not to include DC Offset as part of Automatic port extension.
AutoPortExtLoss	IFixturing2	Specifies whether or not to include loss correction as part of Automatic Port Extension.
AutoPortExtSearchStart	IFixturing2	Set the start frequency for custom user span.

<u>AutoPortExtSearchStop</u>	IFixturing2	Set the stop frequency for custom user span.
<u>AutoPortExtState</u>	IFixturing2	Enables and disables automatic port extensions on the specified port.
<u>CmnModeZConvPortImag</u>	IFixturing2	Sets imaginary value for common port impedance conversion.
<u>CmnModeZConvPortReal</u>	IFixturing2	Sets real value for common port impedance conversion.
<u>CmnModeZConvState</u>	IFixturing2	Turns ON/OFF common port impedance conversion.
<u>CmnModeZConvPortZ0</u>	IFixturing2	Sets impedance value for common port impedance conversion.
<u>DiffPortMatch_C</u>	IFixturing2	Sets Capacitance value of the differential matching circuit.
<u>DiffPortMatch_G</u>	IFixturing2	Sets Conductance value of the differential matching circuit.
<u>DiffPortMatch_L</u>	IFixturing2	Sets Inductance value of the differential matching circuit.
<u>DiffPortMatch_R</u>	IFixturing2	Sets Resistance value of the differential matching circuit.
<u>DiffPortMatchMode</u>	IFixturing2	Sets type of circuit to embed.
<u>DiffPortMatchUserFilename</u>	IFixturing2	Specifies the 4-port touchstone file for user-defined differential matching circuit.
<u>DiffPortMatchState</u>	IFixturing2	Turns ON/OFF differential matching circuit function.
<u>DiffZConvPortImag</u>	IFixturing2	Sets imaginary value for differential port impedance conversion.
<u>DiffZConvPortReal</u>	IFixturing2	Sets real value for differential port impedance conversion.
<u>DiffZConvPortZ0</u>	IFixturing2	Sets impedance value for differential port impedance conversion.
<u>DiffZConvState</u>	IFixturing2	Turns ON/OFF differential port impedance conversion.
<u>Embed4PortA</u>	IFixturing2	Returns PNA portA connections.
<u>Embed4PortB</u>	IFixturing2	Returns PNA portB connections.

<u>Embed4PortC</u>	IFixturing2	Returns PNA portC connections.
<u>Embed4PortD</u>	IFixturing2	Returns PNA portD connections.
<u>Embed4PortList</u>	IFixturing2	Specifies all PNA port connections.
<u>Embed4PortNetworkFilename</u>	IFixturing2	Specifies *.s4p filename.
<u>Embed4PortNetworkMode</u>	IFixturing2	Specify embed, de-embed, or none.
<u>Embed4PortState</u>	IFixturing2	Turns ON or OFF 4-port Network Embed/De-embed.
<u>Embed4PortTopology</u>	IFixturing2	Specifies the PNA / DUT topology.
<u>EnablePowerCompensation</u>	IFixturing5	Compensates source power for combined loss through all fixturing functions.
<u>EnableSnPDataExtrapolation</u>	IFixturing6	Turns ON and OFF SNP file extrapolation for both 2-port and 4-port embedding/de-embedding.
<u>FixturingState</u>	IFixturing	Turns Fixturing ON and OFF on this channel.
<u>NetworkPortMapA</u>	IFixturing6	Read the port mapping of in A port for a 4-port SNP file to be embedded.
<u>NetworkPortMapB</u>	IFixturing6	Read the port mapping of in B port for a 4-port SNP file to be embedded.
<u>NetworkPortMapC</u>	IFixturing6	Read the port mapping of out A port for a 4-port SNP file to be embedded.
<u>NetworkPortMapD</u>	IFixturing6	Read the port mapping of out B port for a 4-port SNP file to be embedded.
<u>Port2PdeembedCktModel</u>	IFixturing	Sets and returns the 2 port De-embedding circuit model for the specified port number.
<u>Port2PdeembedState</u>	IFixturing	Turns 2 port de-embedding ON and OFF on this channel.
<u>PortArbzImag</u>	IFixturing3	Sets and returns the imaginary impedance value for the specified single-ended port number.
<u>PortArbzReal</u>	IFixturing3	Sets and returns the real impedance value for the specified single-ended port number.
<u>PortArbzState</u>	IFixturing	Turns single-ended port impedance ON and OFF on the specified channel.
<u>PortArbzZ0</u>	IFixturing3	Sets and returns the real and imaginary impedance value for the specified single-ended port number.

<u>PortCoupleToSystemMedia</u>	IFixturing4	Couples to system Media type
<u>PortCoupleToSystemVelocity</u>	IFixturing4	Couples to system Velocity Factor
<u>PortDelay</u>	IFixturing	Sets and returns the Port Delay value for the specified port number.
<u>PortDistance</u>	IFixturing4	Sets Port Ext in distance
<u>PortDistanceUnit</u>	IFixturing4	Sets distance units
<u>PortExtState</u>	IFixturing	Turns Port Extension ON and OFF on this channel.
<u>PortExtUse1</u>	IFixturing	Sets and returns the USE1 ON/OFF state for the Loss1 and Freq1 values for the specified port number.
<u>PortExtUse2</u>	IFixturing	Sets and returns the USE2 ON/OFF state for the Loss2 and Freq2 values for the specified port number.
<u>PortFreq1</u>	IFixturing	Sets and returns the 1st Port Frequency value for the specified port number.
<u>PortFreq2</u>	IFixturing	Sets and returns the 2nd Port Frequency value for the specified port number.
<u>PortLoss1</u>	IFixturing	Sets and returns the 1st Port Loss value for the specified port number.
<u>PortLoss2</u>	IFixturing	Sets and returns the 2nd Port Loss value for the specified port number.
<u>PortLossDC</u>	IFixturing	Sets and returns the Port Loss at DC value for the specified port number.
<u>PortMatching_C</u>	IFixturing	Sets and returns the Capacitance, 'C' value for the specified port number.
<u>PortMatching_G</u>	IFixturing	Sets and returns the Conductance, 'G' value for the specified port number.
<u>PortMatching_L</u>	IFixturing	Sets and returns the Inductance, 'L' value for the specified port number.
<u>PortMatching_R</u>	IFixturing	Sets and returns the Resistance, 'R' value for the specified port number.
<u>PortMatchingCktModel</u>	IFixturing	Sets and returns the Port Matching circuit model for the specified port number.

<u>PortMatchingState</u>	IFixturing	Turns Port Matching ON and OFF on this channel.
<u>PortMedium</u>	IFixturing4	Sets Media per port
<u>PortVelocityFactor</u>	IFixturing4	Set Velocity Factor per port
<u>PortWGCutoffFreq</u>	IFixturing4	Sets waveguide cutoff frequency per port
<u>Reverse2PortAdapter</u>	IFixturing6	Set and read whether or not to reverse ports on a 2-port fixture or adapter to be de-embedded.
<u>strPort2Pdeembed_S2PFile</u>	IFixturing	Sets and returns the 2 port De-embedding 'S2P' file name for the specified port number.
<u>strPortMatch_S2PFile</u>	IFixturing	Sets and returns the Port Matching 'S2P' file name for the specified port number.

IFixturing History

Interface	Introduced with PNA Rev:
IFixturing	5.0
IFixturing2	5.2
IFixturing3	5.25
IFixturing4	8.50
IFixturing5	9.20
IFixturing6	9.33

FOM Collection

Description

The FOM collection provides access to the source and receiver range objects which are used for configuring frequency offset measurements.

The FOM range items are typically numbered as follows:

1. Primary
2. Source
3. Receivers
4. Source2 (if present)

External devices can appear in the list of range names. [Learn more.](#)

Accessing the FOM Collection and [FOMRange](#) objects

```
Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel

Dim ifom as FOM
Set ifom = chan.FOM

ifom.item(2).Coupled = false
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [About FOM](#)
- [Example Programs](#)

Method	Interface	Description
	See History	

[Item](#) IFOM

Property	Interface	Description
----------	-----------	-------------

DisplayRange	IFOM	Sets the range to be displayed on the PNA x-axis.
FOMRange	IFOM	Object
RangeCount	IFOM	Returns the number of FOM ranges available on the PNA.
State	IFOM	Turns Frequency Offset ON and OFF.

FOM History

Interface	Introduced with PNA Rev:
IFOM	7.10

Last Modified:

8-Mar-2007 Modified Access

FOMRange Object

Description

The FOM Range object provides access to the properties and methods for configuring a specific Range for frequency offset measurements.

Accessing an FOMRange object

Get a handle to a FOM Range by specifying an item in the [FOM collection](#).

The FOM range items are typically numbered as follows:

1. Primary
2. Source
3. Receivers
4. Source2 (if present)

External devices can appear in the list of range names. [Learn more](#).

```
Dim app as AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim ranges as FOM
Set ranges = app.ActiveChannel.FOM
ranges.item(2).Coupled = False
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [About FOM](#)
- [Example Programs](#)

Method	Interface	Description
	See History	

None

Property	Interface	Description
Coupled	IFOMRange	Sets and returns the state of coupling (ON or OFF) of this range to the primary range.
CWFrequency	IFOMRange	Set the Continuous Wave (CW) frequency.
Divisor	IFOMRange	Sets and returns the Divisor value to be used when coupling this range to the primary range.
Multiplier	IFOMRange	Sets and returns the Multiplier value to be used when coupling this range to the primary range.
Name	IFOMRange	Returns the name of this FOM range object.
Offset	IFOMRange	Sets and returns the offset value to be used when coupling this range to the primary range.
rangeNumber	IFOMRange	Returns the index number of the range within the FOM collection .
Segments	IFOMRange	Collection - Used to add segment sweep capability to a range.
StartFrequency	IFOMRange	Sets or returns the start frequency of this FOM Range.
StopFrequency	IFOMRange	Sets or returns the stop frequency of this FOM Range.
Sweep Type	IFOMRange	Sets the type of range sweep.

Note: Use the [Start Power](#) and [Stop Power](#) settings from the [channel object](#).

FOM History

Interface	Introduced with PNA Rev:
IFOM	7.10

Last Modified:

7-Jan-2008 Added Start/Stop power note
7-Mar-2007 Modified Receivers

Gain Compression Measurement Object

Description

Controls the Gain Compression Analysis settings.

Accessing the GainCompressionMeas object

```
Set app = CreateObject("AgilentPNA835x.Application")
app.Preset
app.ActiveMeasurement.Delete
' create a GCA measurement
app.CreateCustomMeasurementEx 1, "Gain Compression","CompIn21", 1
Set meass = app.Measurements ' get the measurements collection
Set ana = meass(1).CustomMeasurementConfiguration 'get the measurement
ana.AnalysisEnable = true ' enable the analysis mode
```

See Also:

- **Example Program** [Create and Cal a Gain Compression Measurement](#) (includes Analysis)
- [GainCompression Object](#)
- [GainCompressionCal Object](#)
- [About Gain Compression Application](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)

Method	Interface	Description
	See History	

None

Property	Interface	Description
AnalysisCWFreq	IGainCompressionMeas	Set CW frequency.
AnalysisEnable	IGainCompressionMeas	Enable a compression analysis trace.
AnalysisIsDiscreteFreq	IGainCompressionMeas	Set to discrete or interpolated CW frequencies.
AnalysisXAxis	IGainCompressionMeas	Sets X-axis display.

IGainCompressionMeas History

Interface	Introduced with PNA Rev:
IGainCompressionMeas	9.0

Last Modified:

3-Sep-2009 MX New topic

Gain Compression Object

Description

Controls the Gain Compression Application settings.

Accessing the GainCompression object

```
Dim app as AgilentPNA835x.Application
app.CreateCustomMeasurementEx(1, "Gain Compression", "CompIn21", 1)

Dim GCA

Set GCA = app.ActiveChannel.CustomChannelConfiguration
```

See Also:

- **Example Programs**
 - [Create and Cal a Gain Compression Measurement](#)
 - [Create and Cal a GCX Measurement](#)
- [GainCompressionCal Object](#)
- [About Gain Compression Application](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)

Note: Set the Start/Stop Frequency and Start/Stop Power Settings using the [Channel Object](#).

Method	Interface	Description
	See History	
GetRaw2DData	IGainCompression	Reads Gain Compression data from specified location.
GetDataIm	IGainCompression	Reads Imaginary part of specified frequency or power points.
GetDataRe	IGainCompression	Reads REAL part of specified frequency or power points.
SetPortMap	IGainCompression	Maps the PNA ports to the DUT ports

Property	Interface	Description
<u>AcquisitionMode</u>	IGainCompression	Set and read the method by which gain compression data is acquired.
<u>CompressionAlgorithm</u>	IGainCompression	Set and read the algorithm method used to compute gain compression.
<u>CompressionBackoff</u>	IGainCompression	Set and read value for the BackOff compression algorithm.
<u>CompressionDeltaX</u>	IGainCompression	Set and read the 'X' value in the delta X/Y compression algorithm.
<u>CompressionDeltaY</u>	IGainCompression	Set and read the 'Y' value in the delta X/Y compression algorithm.
<u>CompressionInterpolation</u>	IGainCompression	Sets whether or not to interpolate the final power level when the measured compression level deviates from the specified level.
<u>CompressionLevel</u>	IGainCompression	Set and read the decrease in gain which indicates that the amplifier is compressing.
<u>DeviceInputPort</u>	IGainCompression	Read the PNA port number which is connected to the DUT input.
<u>DeviceOutputPort</u>	IGainCompression	Read the PNA port number which is connected to the DUT output.
<u>EndOfSweepOperation</u>	IGainCompression	Set and read the action which should be taken at the end of the last frequency or power sweep in the measurement.
<u>InputLinearPowerLevel</u>	IGainCompression	Set and read the input power level that should produce linear gain.
<u>MaximumNumberOfPoints</u>	IGainCompression	Returns the maximum possible number of data points.
<u>NumberOfFrequencyPoints</u>	IGainCompression	Set and read the number of data points in each frequency sweep.
<u>NumberOfPowerPoints</u>	IGainCompression	Set and read the number of data points in each power sweep.
<u>ReverseLinearPowerLevel</u>	IGainCompression	Set and read the reverse power level to the DUT.

<u>SafeSweepCoarsePowerAdjustment</u>	IGainCompression	Set and read the Safe Sweep COARSE power adjustment.
<u>SafeSweepEnable</u>	IGainCompression	Set and read the (ON OFF) state of Safe Sweep mode.
<u>SafeSweepFinePowerAdjustment</u>	IGainCompression	Set and read the Safe Sweep FINE power adjustment.
<u>SafeSweepFineThreshold</u>	IGainCompression	Set and read the compression level in which Safe Sweep changes from the COARSE power adjustment to the FINE power adjustment.
<u>SafeSweepMaximumLimit</u>	IGainCompression4	When the DUT Output reaches this value, the input power to the DUT is no longer incremented at that frequency.
<u>SaturationLevel</u>	IGainCompression3	Set and read the deviation dB from the maximum Pout.
<u>SearchFailures</u>	IGainCompression	Read number of points that did not achieve compression.
<u>SearchSummary</u>	IGainCompression	Returns if a compression search is complete, and if the search was a success or failure.
<u>SmartSweepMaximumIterations</u>	IGainCompression	Set and read the maximum number of iterations to be used to find the compression level in a SMART sweep.
<u>SmartSweepSettlingTime</u>	IGainCompression	Set and read SMART sweep settling time.
<u>SmartSweepShowIterations</u>	IGainCompression	Set and read whether to show results for each SMART sweep iteration.
<u>SmartSweepTolerance</u>	IGainCompression	Set and read the level of tolerance to be used to find the compression level in a SMART sweep.
<u>TotalIterations</u>	IGainCompression2	Returns the total number of iteration required for the SMART Sweep.
<u>TotalNumberOfPoints</u>	IGainCompression	Set and read the total number of data points.(Freq x Power)

IGainCompression History

Interface	Introduced with PNA Rev:
IGainCompression	8.0
IGainCompression2	8.2
IGainCompression3	9.0
IGainCompression4	9.2

Last Modified:

3-Sep-2009	Added GC3 (9.0)
10-Jun-2009	Added SearchSummary
12-May-2008	Added 8.2
11-Sep-2007	MX New topic

Gain CompressionCal Object

Description

Sets properties that are unique to a Gain Compression Cal (opt 086).

The remaining commands to perform a GCA Cal use the [Guided Calibration commands](#).

Accessing the GainCompressionCal object

```
Dim app as AgilentPNA835x.Application

Set GCACal = pna.GetCalmanager.CreateCustomCalEx\(channelNum\)

Set GCACalExtension = GCACal.CustomCalConfiguration

GCACalExtension.PowerLevel = 5
```

See Also:

- [Example Program Create and Cal a Gain Compression Measurement](#)
- [GainCompression Object](#)
- [About Gain Compression Application](#)
- [The PNA Object Model](#)
- [PNA Automation Interfaces](#)

Method	Interface	Description
	See History	

None

Property	Interface	Description
PowerLevel	IGainCompressionCal	Set and read the power level at which to perform the Source Power Cal.
PowerSensorCalKitType	IGainCompressionCal2	Set and read the cal kit to be used for calibrating at the port 1 reference plane.
PowerSensorConnectorType	IGainCompressionCal2	Superseded by PowerSensorConnectorType on the GuidedCal Object.

IGainCompressionCal History

Interface	Introduced with PNA Rev:
IGainCompressionCal	8.0
IGainCompressionCal2	8.04

Last Modified:

12-Oct-2009	Superseded property (9.1)
11-Apr-2008	Added 8.04 properties
27-Nov-2007	MX New topic

Gating Object

Description

Contains the methods and properties that control Time Domain Gating.

Accessing the Gating Object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim gate As Gating
Set gate = app.ActiveMeasurement.Gating
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Time Domain Topics](#)
- [Example Programs](#)

Methods

None

Properties

Interface Description

(see
[History](#))

Center	IGating	Sets or returns the Center time. Shared with the Transform Object
CoupledParameters	IGating2	Select Gating parameters to couple
Shape	IGating	Specifies the shape of the gate filter.
Span	IGating	Sets or returns the Span time. Shared with the Transform Object
Start	IGating	Sets or returns the Start time. Shared with the Transform Object

State	IGating	Turns an Object ON and OFF.
Stop	IGating	Sets or returns the Stop time.
		Shared with the Transform Object
Type	IGating	Specifies the type of gate filter used.

History

Interface	Introduced with PNA Rev:
IGating	1.0
IGating2	4.2

GlobalPowerLimit Object

Description

Provides access to the properties and methods for setting power limits for PNA ports.

Accessing the GlobalPowerLimit object

```
Dim app as AgilentPNA835x.Application
Dim gpl as IGlobalPowerLimit
Set gpl = app.GlobalPowerLimit
```

See Also:

- [About GlobalPowerLimit](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)

Methods	Description
None	

Properties	Interface	Description
	See History	
Limit	IGlobalPowerLimit	Sets and returns the power limit for the specified port.
Lock	IGlobalPowerLimit	Enables or disables the ability to change the power limit values through the user interface.
State	IGlobalPowerLimit	Turns GlobalPowerLimit ON and OFF

IGlobalPowerLimit History

Interface	Introduced with PNA Rev:
IGlobalPowerLimit	9.0

Last Modified:

5-Aug-2009 MX New topic

GroupDelayAperture Object

Description

Contains the methods and properties that set Group Delay Aperture.

Accessing the GroupDelayAperture Object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application")

Dim GDAperture As GroupDelayAperture
Set GDAperture = app.ActiveMeasurement.GroupDelayAperture
```

See Also:

- [Group Delay Measurements](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods	Interface	Description
None		
Properties	Description	
Frequency	IGroupDelayAperture	Sets group delay aperture using a fixed frequency range.
Percent	IGroupDelayAperture	Sets group delay aperture using a percent of the channel frequency span.
Points	IGroupDelayAperture	Sets group delay aperture using a fixed number of data points.

IGroupDelayAperture History

Interface	Introduced with PNA Rev:
IGroupDelayAperture	A.09.20

Last Modified:

23-Feb-2010 MX New topic

GuidedCalibration Object

Description

Contains the methods and properties used to perform a Guided Calibration.

Important!

Do **NOT** use commands from the [Calibrator](#) (Unguided calibration) object when performing a Guided calibration. Use **ONLY** the [GuidedCalibration](#) object.

The **ONLY** exception is:

Use [OrientECalModule_Property](#) and [ECalPortMapEx_Property](#) on the [Calibrator Object](#) to specify orientation for both guided and unguided calibrations.

A Guided Calibration must be performed on the Active Channel. To activate a channel, activate any measurement on that channel. Do this using [meas.Activate](#), which requires you already have a handle to the measurement.

Accessing the GuidedCalibration object

For standard S-parameter channels:

```
Dim app as AgilentPNA835x.Application
Dim CalMgr
Set CalMgr = App.GetCalManager
Dim guidedCal
Set guidedCal = CalMgr.GuidedCalibration
```

To calibrate an [Application](#) channel, see [CreateCustomCalEx](#) Method.

THRU Pairs sequence

The Smart/Guided Cal logic always determines the best calibration based on your specified connectors and ports.

The following three commands overwrite the SmartCal logic. Send these commands ONLY if you have a deliberate reason for overwriting the SmartCal logic.

- [ThruPortList](#)
- [PathThruMethod](#)
- [PathCalMethod](#)

When sending one or more of these commands, they must be sent in the following sequence with the other commands listed here.

Note: The [Initialize](#) command is sent before and after these three commands.

1. [ConnectorType](#) (ports)
2. [CalKitType](#) (ports)
3. [Initialize](#)
4. [ThruPortList](#)
5. [PathThruMethod](#)
6. [PathCalMethod](#)
7. calMethod = guidedCal.[PathCalMethod](#) (ports) (recommended)
8. [Initialize](#)

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods

Description

[AcquireStep](#)

IGuidedCalibration

Acquire data for a cal step.

ApplyDeltaMatchFromCalSet Method	IGuidedCalibration2	Apply a cal as Delta Match Cal.
GenerateErrorTerms	IGuidedCalibration	Generates the error terms for the calibration.
GenerateGlobalDeltaMatchSequence	IGuidedCalibration2	Initiates a global delta match calibration.
GenerateSteps	IGuidedCalibration	Request to generate a connection list and return the number of steps required.
GetCompatibleCalKits	IGuidedCalibration5	Returns the list of cal kits for the connector type.
GetIsolationPaths	IGuidedCalibration3	Gets the list of port pairings for which isolation standards will be measured during calibration.
GetStepDescription	IGuidedCalibration	Query description of a step.
Initialize	IGuidedCalibration	Initial setup with channel context for the remote cal object.
ResetStep	IGuidedCalibration10	
SetIsolationPaths	IGuidedCalibration3	Sets the list of port pairings for which isolation standards will be measured during calibration.
SetupMeasurementsForStep	IGuidedCalibration4	Show the Cal Window, or custom Cal Window, before acquiring a Cal standard.

Properties

Interface

Description

[See History](#)

CalKitType	IGuidedCalibration	Sets the cal kit for the port.
CompatibleCalKits	IGuidedCalibration	Superseded with GetCompatibleCalKits Method
ConnectorType	IGuidedCalibration	Sets the connector type for the port.
CustomCalConfiguration	IGuidedCalibration4	Provides access to additional Properties and Methods which extends the GuidedCal Object.

IsolationAveragingIncrement	IGuidedCalibration3	Value by which to increment the channel's averaging factor during measurement of isolation standards.
IterationCountForStep	IGuidedCalibration10	
MinimumIterationsForStep	IGuidedCalibration10	
PathCalMethod	IGuidedCalibration3	Specifies the calibration method for each port pair.
PathThruMethod	IGuidedCalibration3	Specifies the calibration THRU method for each port pair.
PerformPowerCalibration	IGuidedCalibration7	Perform Guided Power Cal
PortsNeedingDeltaMatch	IGuidedCalibration2	Returns port numbers that need delta match cal.
PowerCalibrationPowerLevel	IGuidedCalibration6	Sets power level for power cal in several applications.
PowerSensorCalKitType	IGuidedCalibration6	Sets Cal Kit for power cal in several applications.
PowerSensorConnectorType	IGuidedCalibration6	Sets Power sensor connector for power cal in several applications.
SlidingLoadAcquisitionBehavior	IGuidedCalibration10	
ThruCalMethod	IGuidedCalibration	Superseded with PathCalMethod and PathThruMethod
ThruPortList	IGuidedCalibration	Sets the thru connection port pairs.
UseCalWindow	IGuidedCalibration	Obsolete - Use Custom Cal Window commands.
ValidConnectorTypes	IGuidedCalibration	Gets Valid Connector Types.

IGuidedCalibration History

Interface	Introduced with PNA Rev:
IGuidedCalibration	5.0
IGuidedCalibration2	5.25
IGuidedCalibration3	7.11
IGuidedCalibration4	8.0
IGuidedCalibration5	9.0
IGuidedCalibration6	9.10
IGuidedCalibration7	9.30
IGuidedCalibration8	
IGuidedCalibration9	
IGuidedCalibration10	9.85

Last Modified:

- 4-Dec-2012 Added sliding load commands (BH)
- 27-Jan-2012 Obsolete UseCalWindow
- 5-Jan-2011 Added Calibrator note
- 8-Oct-2010 Fixed PowerCalLevel
- 9-Sep-2010 Added IGuidedCalibration7
- 25-Aug-2009 Added GetCompatibleCalKits
- 22-Jul-2009 Added CustomCalConfiguration
- 9-Nov-2007 Added Setup command and Activate note

GuidedCalibrationPowerSensor Object

Description

Contains the methods and properties to configure multiple power sensors to be used during a guided power calibration.

Note: These commands are supported ONLY on standard channels.

Accessing the GuidedCalibrationPowerSensor Object

A GuidedCalibrationPowerSensor object is accessed as an item of the [GuidedCalibrationPowerSensors Collection](#).

```
Option explicit

Dim app as AgilentPNA835x.Application

Set app = CreateObject("AgilentPNA835x.Application")

Dim CalMgr

Set CalMgr = App.GetCalManager

Dim guidedCal

Set guidedCal = CalMgr.GuidedCalibration

Dim powerSensors

Dim port: port = 1

Set powerSensors = guidedCal.GuidedCalibrationPowerSensors(port)

Dim powerSensor2

Set powerSensor2 = GuidedCalibrationPowerSensors.Item(2)

powerSensor2.Name="26GHzSensor"
```

See Also:

- [GuidedCalibrationPowerSensors Collection](#)
- See Example Programs
 - [Perform a Guided Power Cal using Multiple Power Sensors](#)
 - [Create a PMAR Device and Measurement](#)

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)

Methods	Description
---------	-------------

None

Properties	Interface	Description
------------	-----------	-------------

[See History](#)

Name	IGuidedCalibrationPowerSensors	Sets and returns the name of this power sensor.
PowerSensorCalKitType	IGuidedCalibrationPowerSensors	Set and read the cal kit to be used for calibrating at the reference plane.
PowerSensorConnectorType	IGuidedCalibrationPowerSensors	Sets or returns the connector type of the power sensor
StartFrequency	IGuidedCalibrationPowerSensors	Sets or returns the start frequency of the power sensor coverage
StopFrequency	IGuidedCalibrationPowerSensors	Sets or returns the stop frequency of the power sensor coverage

IGuidedCalibrationPowerSensors History

Interface	Introduced with PNA Rev:
IGuidedCalibrationPowerSensors	9.33

Last Modified:

16-Jun-2011 Updated example and links
 29-Nov-2010 New topic

GuidedCalibrationPowerSensors Collection

Description

Contains the methods and properties to enable and configure multiple power sensors to be used during a guided calibration.

Note: Guided Power Cal, and "multiple sensors" are allowed ONLY on standard channels.

Accessing the GuidedCalibrationPowerSensors Collection

```
Option explicit

Dim app as AgilentPNA835x.Application

Set app = CreateObject("AgilentPNA835x.Application")

Dim CalMgr

Set CalMgr = App.GetCalManager

Dim guidedCal

Set guidedCal = CalMgr.GuidedCalibration

Dim port: port = 1

Dim powerSensors

Set powerSensors = guidedCal.GuidedCalibrationPowerSensors(port)

powerSensors = powerSensors.Add "26GHzPowerSensor"
```

See Also:

- [GuidedCalibrationPowerSensor Object](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [See Example Program](#)

Methods	Description
---------	-------------

Add	IGuidedCalibrationPowerSensors Add a power sensor by name to the collection.
Item	IGuidedCalibrationPowerSensors Specify a power sensor by name in the collection.
Remove	IGuidedCalibrationPowerSensors Remove a power sensor from the collection.

Properties	Interface	Description
------------	-----------	-------------

[See History](#)

Count	IGuidedCalibrationPowerSensors	Return the number of power sensors in the collection.
UseMultipleSensors	IGuidedCalibrationPowerSensors	Enable the use of multiple power sensors.

IGuidedCalibrationPowerSensors History

Interface	Introduced with PNA Rev:
IGuidedCalibrationPowerSensors	9.33

Last Modified:

29-Nov-2010 New topic

HWAuxIO Object

Description

Contains the methods and properties that control the rear panel Material Handler I/O and Power I/O connector.

Note: The Aux I/O connector is mentioned in these topics, but NO PNA models supported by this PNA firmware have that connector.

Accessing the HWAuxIO object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim AuxIO As HWAuxIO
Set AuxIO = app.GetAuxIO
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods	Interface	Description
	See History	
get_InputVoltage	IHWAuxIO	Superseded by get InputVoltageEX
get_OutputVoltage	IHWAuxIO	Reads ADC output voltages.
get_OutputVoltageMode	IHWAuxIO2	Reads mode setting for either DAC output.
get_PortCData	IHWAuxIO	Reads a 4-bit value from Port C
put_OutputVoltage	IHWAuxIO	Writes voltages to the DAC/Analog Output 1 and Output 2
put_OutputVoltageMode	IHWAuxIO2	Writes the mode setting for either DAC output.
put_PortCData	IHWAuxIO	Writes a 4-bit value to Port C
Properties		Description
FootSwitch	IHWAuxIO	Obsolete - Reads the Footswitch Input

FootswitchMode	IHWAuxIO3	Obsolete -Determines the action that occurs when the footswitch is pressed.
InputVoltageEX	IHWAuxIO5	Reads the ADC input voltage
PassFailLogic	IHWAuxIO	Sets and reads the logic of the PassFail line Shared with the HWMaterialHandler Object
PassFailMode	IHWAuxIO	Sets and reads the mode of the PassFail line Shared with the HWMaterialHandler Object
PassFailPolicy	IHWAuxIO4	Sets the policy used to determine how global pass/fail is computed. Shared with the HWMaterialHandler Object
PassFailScope	IHWAuxIO	Sets and reads the scope of the PassFail line Shared with the HWMaterialHandler Object
PassFailStatus	IHWAuxIO4	Returns the most recent pass/fail status value. Shared with the HWMaterialHandler Object
PortCLogic	HWAuxIO	Sets and reads the logic mode of Port C
PortCMode	HWAuxIO	Sets and reads the mode of Port C
SweepEndMode	HWAuxIO	Sets and reads the event that causes the Sweep End line to go to a false state. Shared with the HWMaterialHandler Object

IHWAuxIO History

Interface	Introduced with PNA Rev:
IHWAuxIO	2.0
IHWAuxIO2	3.0
IHWAuxIO3	3.0
IHWAuxIO4	5.0
IHWAuxIO5	7.5

Last Modified:

5-Aug-2008 Updated for InputVoltageEX command
10-Jul-2007 Added new command
29-Jun-2007 Updated for PNA-X ADC commands

HWExternalTestSetIO Object

Description

Contains the methods and properties that control the rear panel External Test Set Input / Output connector

Accessing the HWExternalTestSetIO object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim ExtTS As HWExternalTestSetIO
Set ExtTS = app.GetExternalTestSetIO
```

See Also:

- [Pinout for the External Test Set Connector](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods	Description
ReadData	Reads data and generates the appropriate timing signals
ReadRaw	Reads data, but does NOT generate appropriate timing signals
WriteData	Writes data and generates the appropriate timing signals
WriteRaw	Writes data, but does NOT generate the appropriate timing signals

Properties	Description
Interrupt	Returns the state of the Interrupt line
SweepHoldOff	Returns the state of the Sweep Holdoff line

IHWExternalTestSetIO History

Interface	Introduced with PNA Rev:
------------------	-------------------------------------

HWExternalTestSetIO	2.0
---------------------	-----

HWMaterialHandlerIO Object

Description

Contains the methods and properties that control the rear panel Material Handler Input / Output connector.

Accessing the HWMaterialHandlerIO object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim MatHdlr As HWMaterialHandlerIO
Set MatHdlr = app.GetMaterialHandlerIO
```

See Also:

- [Pinout for the Material HandlerIO Connector](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods	Interface	Description
get_Input1	HWMaterialHandlerIO	Reads a hardware latch that captures low to high transition on Input1
get_Output	HWMaterialHandlerIO	Returns the last value written to the selected output pin.
get_Port	HWMaterialHandlerIO	Returns the value from the specified "readable" port.
put_Output	HWMaterialHandlerIO	Writes a TTL HI or TTL Low to output pins 3 or 4.
put_Port	HWMaterialHandlerIO	Writes a value to the specified port.
Properties		Description
IndexState	HWMaterialHandlerIO2	Determines the control of Material Handler connector Pin 20

<u>ReadyForTriggerState</u>	HWMaterialHandlerIO2	Determines the control of Material Handler connector Pin 21
<u>PassFailLogic</u>	HWMaterialHandlerIO	Sets and reads the logic of the PassFail line Shared with the HWAuxIO Object
<u>PassFailMode</u>	HWMaterialHandlerIO	Sets and reads the mode for the PassFail line Shared with the HWAuxIO Object
<u>PassFailPolicy</u>	HWMaterialHandlerIO2	Sets the policy used to determine how global pass/fail is computed. Shared with the HWAuxIO Object
<u>PassFailScope</u>	HWMaterialHandlerIO	Sets and reads the scope for the PassFail line Shared with the HWAuxIO Object
<u>PassFailStatus</u>	HWMaterialHandlerIO2	Returns the most recent pass/fail status value. Shared with the HWAuxIO Object
<u>PortLogic</u>	HWMaterialHandlerIO	Sets and returns the logic mode of data ports A-H
<u>PortMode</u>	HWMaterialHandlerIO	Sets and returns whether Port C or Port D is used for writing or reading data
<u>SweepEndMode</u>	HWMaterialHandlerIO	Sets and reads the event that cause the Sweep End line to go to a low state. Shared with the HWAuxIO Object

HWMaterialHandlerIO History

Interface	Introduced with PNA Rev:
HWMaterialHandlerIO	2.0
HWMaterialHandlerIO2	5.0

IFConfiguration Object

Description

These properties control the IF gain and source path settings for the PNA-X ([IFConfiguration3](#) commands ONLY).

Accessing the IFConfiguration object

```
Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel
Dim cfg as IIFConfiguration
Set cfg = chan.IFConfiguration
```

See Also

- [IF Path Configuration](#)
- [SignalProcessingModuleFour Object](#) (PNA-X ONLY)
- [PulseGenerator Object](#) (PNA-X ONLY)
- [The PNA Object Model](#)
- [Pulsed Application](#)

Methods	Description
---------	-------------

None

Properties	Interface	Description
------------	-----------	-------------

[See History](#)

IFFrequency	IFConfiguration3	Sets IF frequency in manual mode.
IFFrequencyMode	IFConfiguration3	Sets IF frequency mode to automatic or manual.
MaximumIFFrequency	IFConfiguration3	Returns the maximum IF frequency setting
MinimumIFFrequency	IFConfiguration3	Returns the minimum IF frequency setting

IFConfiguration History

Interface	Introduced with PNA Rev:
IIFConfiguration	4.0
IIFConfiguration2	4.0
IIFConfiguration3	7.2

Last modified:

July 26, 2012	Removed IFConfig2 and 3 properties
---------------	------------------------------------

IMixer Interface (Option 083) - Superseded

Note: This object and all properties and methods are replaced with Converter Object which can be used for all converter application.

Description

Contains the methods and properties to setup FCA Mixer measurements.

For performing calibrations, use either the SMC Type Object or the VMC Type Object .

Accessing the IMixer Interface

Access the IMixer Interface through the Measurement Object. If the particular type of Measurement that was created supports IMixer, then the program determines this at run time and can access the functionality exposed by IMixer. Because the determination of IMixer support is not made until runtime, the program should handle the case where IMixer is not supported on the object.

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", "analyzerName ")
app.Preset

' FCA Measurements can't share the channel with standard measurements
' Because preset creates a single measurement in channel 1, we first delete the
standard measurement
Dim standardMeas As IMeasurement
Set standardMeas = app.ActiveMeasurement
standardMeas.Delete

' Create a Measurement object, in this case using the IMeasurement interface
Dim meas As IMeasurement
Set meas = app.CreateCustomMeasurementEx(1, "SMC_Forward.SMC_ForwardMeas", "SC21")

' See if this measurement object supports IMixer
Dim mixer As IMixer
Set mixer = meas
```

See Also:

PNA Automation Interfaces

The PNA Object Model

Example: How to create and calibrate a standard SMC or VMC measurement or a fixed output SMC measurement.

Methods	Interface	Description
	See History	

Apply	IMixer3	Applies mixer settings.
Calculate	IMixer	Automatically calculate Input and Output frequencies for mixer setup.
LoadFile	IMixer	Loads a previously-configured mixer attributes file (.mxr)
SaveFile	IMixer	Saves the settings for the mixer/converter test setup to a mixer attributes file.
SetDutPorts	IMixer8	Sets the PNA to DUT port mapping.

Properties	Description
------------	-------------

ActiveXAxisRange	IMixer3	Sets or returns the swept parameter to display on the X-axis.
AvoidSpurs	IMixer	Sets and returns the state of the avoid spurs feature.
EmbeddedLO	IMixer7	Provides measurements of mixers with an embedded LO.
DeviceInputPort	IMixer8	Reads the PNA port that is mapped to the DUT Input port
DeviceOutputPort	IMixer8	Reads the PNA port that is mapped to the DUT Output port
EnablePhase	IMixer12	Include phase in SMC measurements and calibrations.
IFSideband	IMixer	Sets or returns the value of the IF sideband.
IFStartFrequency	IMixer	Returns the start frequency of the mixer IF.
IFStopFrequency	IMixer	Returns the stop frequency of the mixer IF.
IncludeReverseSweep	IMixer12	Sets or returns whether to include SC12 sweep.
InputDenominator	IMixer	Sets or returns the denominator value of the Input Fractional Multiplier.

InputFixedFrequency	IMixer6	Sets or returns the mixer fixed Input frequency value.
InputNumerator	IMixer	Sets or returns the numerator value of the Input Fractional Multiplier.
InputPower	IMixer	Sets or returns the value of the Input Power.
InputRangeMode	IMixer6	Sets or returns the Input sweep mode.
InputStartFrequency	IMixer	Sets or returns the start frequency of the mixer input.
InputStopFrequency	IMixer	Sets or returns the stop frequency of the mixer input.
IsInputGreaterThanLO	IMixer2	Specifies whether to use the Input frequency that is greater than the LO or less than the LO.
LODenominator	IMixer	Sets or returns the denominator value of the LO Fractional Multiplier.
LOFixedFrequency	IMixer	Sets or returns the fixed frequency of the specified LO.
LOName	IMixer	Sets or returns the LO name.
LONumerator	IMixer	Sets or returns the numerator value of the LO Fractional Multiplier.
LOPower	IMixer	Sets or returns the value of the LO Power.
LORangeMode	IMixer3	Sets or returns the LO sweep mode to fixed or swept.
LOStage	IMixer	Returns the number of LOs (1 or 2).
LOStartFrequency	IMixer3	Sets or returns the start frequency of the specified LO.
LOStopFrequency	IMixer3	Sets or returns the start frequency of the specified LO.
NominalIncidentPowerState	IMixer4	Toggles Nominal Incident Power ON and OFF.
NormalizePoint	IMixer12	Set or return the data point used to normalize SMC phase measurements.
OutputFixedFrequency	IMixer3	Sets or returns the fixed frequency of the mixer output.

OutputRangeMode	IMixer6	Sets or returns the Output sweep mode.
OutputSideband	IMixer	Sets or returns the value of the output sideband.
OutputStartFrequency	IMixer	Sets or returns the start frequency of the mixer output.
OutputStopFrequency	IMixer	Sets or returns the stop frequency of the mixer output.

IMixer History

Interface	Introduced with PNA Rev:
IMixer	1.0
IMixer2	3.5
IMixer3	4.0
IMixer4	4.8
IMixer5	6.04
IMixer6	6.20
IMixer7	7.21
IMixer8	7.5/8.2
IMixer12	7.5/9.2

IMSpectrum Object

Description

Controls the IM Spectrum settings.

Accessing the IMSpectrum object

```
Dim app as AgilentPNA835x.Application  
app.CreateCustomMeasurementEx(1, "IMSpectrum", "Output", 1)  
  
Dim IMSpec  
  
Set IMSpec = app.ActiveChannel.CustomChannelConfiguration
```

See Also:

- **Example Program** Create and Cal a IM Spectrum Measurement
- [About IM Spectrum Measurements](#)
- [SweptIMD Object](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)

Method	Interface	Description
	See History	
SetPortMap	IIMSpectrum	Sets limited port mapping
Property	Interface	Description
CoupleTonePower	IIMSpectrum	ON OFF state of power coupling for F1 and F2.
DeviceInputPort	IIMSpectrum	Read input port map
DeviceOutputPort	IIMSpectrum	Read output port map
DeltaFrequency	IIMSpectrum	Fixed tone spacing value.
EqualTonePower	IIMSpectrum2	Superseded with LevelingMethod Set Equal Tone Power state.
F1Frequency	IIMSpectrum	Frequency of the F1 tone.

F2Frequency	IIMSpectrum	Frequency of the F2 tone.
FrequencyCenter	IIMSpectrum	Center frequency of the main tones.
LevelingMethod	IIMSpectrum3	Set tone power leveling mode
ResolutionBW	IIMSpectrum	Resolution Bandwidth for the measurement.
SpectrumCenterFrequency	IIMSpectrum	Receiver Center frequency
SpectrumSpanFrequency	IIMSpectrum	Receiver frequency span.
SpectrumStartFrequency	IIMSpectrum	Receiver Start frequency.
SpectrumStopFrequency	IIMSpectrum	Receiver Stop frequency.
TonePowerSetAt	IIMSpectrum2	Superseded with LevelingMethod Set power at DUT input or Output
TrackingChannel	IIMSpectrum	IMD channel number to which the IM Spectrum channel is coupled.
TrackingEnable	IIMSpectrum	Enables tracking with an IMD channel.
TrackingManualStepEnable	IIMSpectrum	Step sweep mode for the IM Spectrum channel.
TrackingStepIndex	IIMSpectrum	IMD data point number at which the IM spectrum measurement occurs.
SweepOrder	IIMSpectrum	IM product to view when SweepType = NTH is specified.
SweepType	IIMSpectrum	Type of sweep for an IIMSpectrum measurement.
TonePower	IIMSpectrum	Power level of the Main Tones

IIMSpectrum History

Interface	Introduced with PNA Rev:
IIMSpectrum	8.35
IIMSpectrum2	9.40

Last Modified:

5-May-2011	Added tone power settings
11-Aug-2009	Added limited port mapping
19-Aug-2008	MX New topic

InterfaceControl Object

Description

Contains the methods and properties that support Interface Control.

Accessing the InterfaceControl object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim IntControl As InterfaceControl
Set IntControl = app.InterfaceControl
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Interface Control Feature](#)
- [Example Programs](#)

Methods	Interface	Description
ConfigurationFile	InterfaceControl	Recalls an Interface Control file
Properties		Description
State	InterfaceControl	Turns Interface Control ON and OFF

InterfaceControl History

Interface	Introduced with PNA Rev:
InterfaceControl	5.2

Limit Test Collection

Description

Child of the **Measurement** Object. A collection that provides a mechanism for iterating through the Measurement's Limit Segment objects (Limit Lines). The collection has 100 limit lines by default.

Accessing the LimitTest collection

Get a handle to an individual limit segment by specifying an item of the LimitTest collection.

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim limSegs As LimitTest
Set limSegs = app.ActiveMeasurement.LimitTest
limSegs.Item(1).BeginStimulus = 1000000000
limSegs.Item(1).EndStimulus = 1000000000
limSegs.Item(1).BeginResponse = 3.5
limSegs.Item(1).EndResponse = 3.5
```

See Also:

- [LimitSegment Object](#)
- [Collections in the Analyzer](#)
- [The PNA Object Model](#)
- [Limit Line Testing Example](#)

Methods	Description
GetTestResult	Retrieves the Pass/Fail results of the Limit Test (State).
Item	Use to get a handle on a limit line in the collection.

Properties	Description
Count	Returns the number of limit lines used in the measurement.
LineDisplay	Displays the limit lines on the screen.
SoundOnFail	Enables a beep on Limit Test fails.
State	Turns ON and OFF limit testing.

LimitTest History

Interface	Introduced with PNA Rev:
ILimitTest	1.0

LimitSegment Object

Description

The LimitSegment object is an individual limit line.

Accessing the LimitSegment object

Get a handle to an individual limit line by using the LimitTest collection.

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim limSegs As LimitTest
Set limSegs = app.ActiveMeasurement.LimitTest
limSegs(1).BeginResponse = 1000000000#
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods	Description
---------	-------------

None

Properties	Description
------------	-------------

[BeginResponse](#) Specifies the Y-axis value that corresponds with Begin Stimulus (X-axis) value.

[BeginStimulus](#) Specifies the beginning X-axis value of the Limit Line.

[EndResponse](#) Specifies the Y-axis value that corresponds with End Stimulus (X-axis) value.

[EndStimulus](#) Specifies the End X-axis value of the Limit Line.

[Type](#) Specifies the Limit Line type.

LimitSegment History

Interface

**Introduced with PNA
Rev:**

ILimitSegment

1.0

Marker Object

Description

Contains the methods and properties that control Markers. There are 10 markers available per measurement:

- 1 reference marker
- 9 markers for absolute data or data relative to the reference marker (delta markers).

There are two ways to control markers through COM.

1. The [Measurement object](#) has properties that apply to ALL of the markers for that measurement. For example, **meas.MarkerFormat = naLinMag** applies formatting to all markers.
2. Marker object properties override the Measurement object properties. For example, you can then override the format setting for an individual marker by specifying **mark.Format = naLogMag** on the marker object.

Note: [SearchFilterBandwidth](#) is available through the [measurement object](#).

Accessing the Marker object

To turn ON a marker, get a handle to the marker through the [measurement object](#). If not already activated, this command will turn ON marker 1

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

app.ActiveMeasurement.marker(1).Format = naLinMag
```

You can also set the marker object to an object variable:

```
Dim m1 As Marker
Set m1 = app.ActiveMeasurement.Marker(1)
m1.Format = naMarkerFormat_LinMag
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)
- [All about Markers](#)

Methods	Interface	Description
Activate	IMarker	Makes an object the Active Object. Shared with the Marker Object
SearchCompressionPoint	IMarker4	Searches the marker domain for the compression level.
SearchMax	IMarker	Searches the marker domain for the maximum value.
SearchMin	IMarker	Searches the marker domain for the minimum value.
SearchNextPeak	IMarker	Searches the marker's domain for the next largest peak value.
SearchPeakLeft	IMarker	Searches the marker's domain for the next VALID peak to the left of the marker.
SearchPeakRight	IMarker	Searches the marker's domain for the next VALID peak to the right of the marker.
SearchTarget	IMarker	Searches the marker's domain for the target value.
SearchTargetLeft	IMarker	Moving to the left of the marker position, searches the marker's domain for the target value.
SearchTargetRight	IMarker	Moving to the right of the marker position, searches the marker's domain for the target value.
SetCenter	IMarker	Changes the analyzer's center frequency to the X-axis position of the marker.
SetCW	IMarker	Changes the sweep type to CW mode and makes the CW frequency the marker's frequency.
SetCWFreq	IMarker3	Changes the CW frequency to the frequency of the active marker.
SetElectricalDelay	IMarker	Changes the measurement's electrical delay to the marker's delay value.
SetReferenceLevel	IMarker	Changes the measurement's reference level to the marker's Y-axis value.
SetStart	IMarker	Changes the analyzer's start frequency to the X-axis position of the marker.
SetStop	IMarker	Changes the analyzer's stop frequency to the X-axis position of the marker.

Properties		Description
Bucket Number	IMarker	Marker data point number
CompressionLevel	IMarker4	Set and read the marker compression level.
CompressionPin	IMarker4	Reads the input power at the marker compression level.
CompressionPout	IMarker4	Reads the output power at the marker compression level.
DeltaMarker	IMarker	Makes a marker relative to the reference marker
Distance	IMarker2	Sets or returns distance value for time domain trace.
Format	IMarker	Linear, SWR, and so forth
Interpolated	IMarker	Turn marker interpolation ON and OFF
Number	IMarker	Read the number of the active marker
PeakExcursion	IMarker	Sets and reads the peak excursion value for the specified marker.
PeakThreshold	IMarker	Sets peak threshold for the specified marker.
SearchFunction	IMarker	Emulates the Tracking function in the marker search dialog box.
Stimulus	IMarker	Sets and reads the X-Axis value of the marker.
Target Value	IMarker	Sets the target value for the marker when doing Target Searches.
Tracking	IMarker	The tracking function finds the selected search function every sweep.
Type	IMarker	Sets and reads the marker type.
UserRange	IMarker	Assigns the marker to the specified User Range.
UserRangeMax	IMarker	Sets the stimulus stop value for the specified User Range.
UserRangeMin	IMarker	Sets the stimulus start value for the specified User Range.
Value	IMarker	Reads the Y-Axis value of the marker.

Marker History

Interface	Introduced with PNA Rev:
------------------	---------------------------------

IMarker	1.0
IMarker2	4.2
IMarker3	8.33
IMarker4	8.50

Measurement Object

See [IArrayTransferInterface](#) for putting and getting typed data.

See [IMixer Interface](#) (used with [Option 083](#))

Description

The Measurement object is probably the most used object in the PNA Object Model. A measurement object represents the chain of data processing algorithms that take raw data from the channel and make it ready for display, which then becomes the scope of the [Trace object](#).

A Measurement object is defined by its parameter (S11, S22, A/R1, B and so forth). The measurement object is associated with a [channel](#) which drives the hardware that produces the data that feeds the measurement. The root of a measurement is the raw data. This buffer of complex paired data then flows through a number of processing blocks: error-correction, trace math, phase correction, time domain, gating, formatting. All of these are controlled through the measurement object.

The ACTIVE measurement is the measurement that will be acted upon if you make a setting from the front panel. It is the measurement whose "button" is pressed in the window with the red "active window" frame. If you create a new measurement, that measurement becomes the active measurement.

Therefore, all automation methods with the word "Active" in them refer to the object associated with the Active measurement, whether that object is a Channel, Window, Trace or Limit line.

Learn about the [IMeasurement2 Interface](#) for reading stimulus properties.

Accessing the Measurement object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim meas As IMeasurement
Set meas = app.ActiveMeasurement
```

or

```
Set meas = app.Measurements(n)
```

You can access four other objects through the Measurement object: markers, limit test, transform, and gating. For example, because each measurement has its own set of markers, you can set a marker by doing this:

```
Dim meas as measurement
Set meas = app.ActiveMeasurement
meas.marker(1).Stimulus = 900e6
```

IMeasurement2 Interface

Some of the properties and methods for the IMeasurement2 Interface return stimulus values that are set using the channel object. The following is the reason these properties and methods are duplicated.

Every measurement carries with it a snapshot of the stimulus properties of the channel that were in effect when the measurement last acquired data. Therefore, it is the measurement that provides the most accurate stimulus description of its data. Any change made to the channel after the measurement was acquired renders the IChannel interface unreliable in terms of describing the measurement.

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)
- [Superseded commands](#)

(**Bold** Methods or Properties provide access to a child object)

Methods	Interface	Description
Activate	IMeasurement	Makes a measurement the active measurement. Shared with the Marker Object
ActivateMarker	IMeasurement	Makes a marker the Active Marker.
ChangeParameter	IMeasurement	Changes the parameter of the measurement.
DataToDivisor	IMeasurement	Superseded with DoReceiverPowerCal Method
DataToMemory	IMeasurement	Stores the active measurement into memory.
Delete	IMeasurement	Deletes the measurement object.
DeleteAllMarkers	IMeasurement	Deletes all of the markers from the measurement.
DeleteMarker	IMeasurement	Deletes a marker from the active measurement.
getData	IMeasurement	Retrieves Complex data from analyzer memory

getDataByString	IMeasurement	Retrieves variant data from the specified location in your choice of formats.
GetFilterStatistics	IMeasurement	Returns all four Filter Statistics
GetReferenceMarker	IMeasurement	Returns a handle to the reference marker.
Get SnPData	IMeasurement3	Returns SnP data.
GetSnPDataWithSpecifiedPorts	IMeasurement7	Returns sNp data for the specified ports.
GetTraceStatistics	IMeasurement	Returns the Trace Statistics.
GetXAxisValues	IMeasurement2	Returns the stimulus values for the measurement.
InterpolateMarkers	IMeasurement	Turns All Marker Interpolation ON and OFF for the measurement.
putDataComplex	IMeasurement	Puts complex data into one of five data buffers.
putDataScalar	IMeasurement	Puts formatted variant data into the measurement results buffer.
SearchFilterBandwidth	IMeasurement	Searches the domain with the current BW target.
WriteSnPFileWithSpecifiedPorts	IMeasurement7	Write sNp data for specified ports to a file.

Properties	Interface	Description
ActiveMarker	IMeasurement	Returns a handle to the Active Marker object.
ActiveXAxisRange	IMeasurement4	Sets the X-axis range to display for GCX, SMC, and VMC measurements.
BalancedMeasurement	IMeasurement5	Sets the measurement type that is used with balanced topologies.
BandwidthTarget	IMeasurement	The insertion loss value at which the bandwidth of a filter is measured.
BandwidthTracking	IMeasurement	Turns Bandwidth Tracking function ON and OFF.
CalibrationName	IMeasurement2	Returns the name of the cal type.
CalibrationType	IMeasurement	Superseded with CalibrationTypeID_property

<u>CalibrationTypeID</u>	IMeasurement2	Sets or returns the cal type for the current measurement.
<u>Center</u>	IMeasurement2	Returns the stimulus value of the center point for the measurement.
<u>channelNumber</u>	IIMeasurement	Returns the channel number. Shared with the Channel Object
<u>CustomMeasurementConfiguration</u>	IIMeasurement12	Provides access to custom measurement properties and methods.
<u>Domain</u>	IMeasurement2	Returns the domain (frequency, time, power) for the measurement.
<u>ElectricalDelay</u>	IMeasurement	Sets electrical delay.
<u>ElecDelayMedium</u>	IMeasurement2	Sets or returns the characteristic of the electrical delay medium.
<u>ElecDistanceDelay</u>	IMeasurement11	Sets delay in distance
<u>ElecDistanceDelayUnit</u>	IMeasurement11	Sets distance units
<u>Equation</u>	IMeasurement6	Access Equation Editor
<u>ErrorCorrection</u>	IMeasurement	Set or get the state of error correction for the measurement.
<u>ErrorCorrectionIndicator</u>	IMeasurement14	Returns the error correction status of the measurement.
<u>FilterBW</u>	IMeasurement	Returns the results of the SearchBandwidth method.
<u>FilterCF</u>	IMeasurement	Returns the Center Frequency result of the SearchBandwidth method.
<u>FilterLoss</u>	IMeasurement	Returns the Loss value of the SearchBandwidth method.
<u>FilterQ</u>	IMeasurement	Returns the Q (quality factor) result of the SearchBandwidth method.
<u>Format</u>	IMeasurement	Sets display format.
<u>FormatUnit</u>	IMeasurement9	Sets units for unratioded measurements.
<u>Gating</u>	IMeasurement	Controls Time Domain Gating.
<u>GroupDelayAperture</u>	IMeasurement13	Provides access to the Group Delay Aperture settings.

InterpolateCorrection	IMeasurement	Turns ON and OFF the calculation of new error terms when stimulus values change.
InterpolateNormalization	IMeasurement	Superseded with DoReceiverPowerCal Method
IsSparameter	IMeasurement2	Returns true if measurement represents an S-Parameter.
LimitTest	IMeasurement	Collection for iterating through the Limit Segment objects (Limit Lines).
LimitTestFailed	IMeasurement	Returns the results of limit testing
LoadPort	IMeasurement	Returns the load port number associated with an S-parameter reflection measurement.
LogMagnitudeOffset	IMeasurement	Superseded with DoReceiverPowerCal Method
MagnitudeOffset	IMeasurment4	Offsets the magnitude of the entire data trace to a specified value.
MagnitudeSlopeOffset	IMeasurment4	Offsets the magnitude of the data trace to a value that changes linearly with frequency.
Marker	IMeasurement	Provides access to Marker settings.
MarkerFormat	IMeasurement	Sets or returns the format of all the markers in the measurement.
Marker State	IMeasurement3	Sets or returns the ON / OFF state of a marker.
Mean	IMeasurement	Returns the mean value of the measurement.
Name	IMeasurement	Sets or returns the name of the measurement.
NA Window	IMeasurement	Controls the part of the display that contains the graticule, or what is written on the display.
Normalization	IMeasurement	Superseded with DoReceiverPowerCal Method
Number	IMeasurement	Returns the number of the measurement.
NumberOfPoints	IMeasurement2	Returns the Number of Points of the measurement.
Parameter	IMeasurement	Returns the measurement Parameter.

<u>PeakToPeak</u>	IMeasurement	Returns the Peak to Peak value of the measurement.
<u>PhaseOffset</u>	IMeasurement	Sets the Phase Offset for the active channel.
PNOP	IMeasurement13	Provides access to the Power Normal Operating Point marker search object.
PSaturation	IMeasurement13	Provides access to the Power Saturation marker search object.
<u>ReceivePort</u>	IMeasurement2	Returns the receiver port of the measurement.
<u>ReferenceMarkerState</u>	IMeasurement	Turns the reference marker ON or OFF
<u>ShowStatistics</u>	IMeasurement	Displays and hides the measurement statistics (peak-to-peak, mean, standard deviation) on the screen.
<u>Smoothing</u>	IMeasurement	Turns ON and OFF data smoothing.
<u>SmoothingAperture</u>	IMeasurement	Specifies or returns the amount of smoothing as a ratio of the number of data points in the measurement trace.
<u>SourcePort</u>	IMeasurement2	Returns the source port of the measurement.
<u>Span</u>	IMeasurement2	Returns the stimulus span (stop - start) for the measurement.
<u>StandardDeviation</u>	IMeasurement	Returns the standard deviation of the measurement.
<u>Start</u>	IMeasurement2	Returns the stimulus value of the first point for the measurement.
<u>StatisticsRange</u>	IMeasurement	Sets the User Range number for calculating measurement statistics.
<u>Stop</u>	IMeasurement2	Returns the stimulus value of the last point for the measurement.
Trace	IMeasurement	Controls scale, reference position, and reference line.
<u>TraceMath</u>	IMeasurement	Performs math operations on the measurement object and the trace stored in memory.

TraceMax	IMeasurement10	Maximizes the active trace.
TraceTitle	IMeasurement8	Writes and reads a trace title.
TraceTitleState	IMeasurement8	Turns trace title ON and OFF
Transform	IMeasurement	Controls Time Domain transforms.
UserRangeMax	IMeasurement15	Sets the stimulus stop value for the specified User Range.
UserRangeMin	IMeasurement15	Sets the stimulus start value for the specified User Range.
View	IMeasurement	Sets (or returns) the type of trace displayed on the screen.
WGCutoffFreq	IMeasurement2	Sets or returns the value of the waveguide cut off frequency.

IMeasurement History

Interface	Introduced with PNA Rev:
IMeasurement	1.0
IMeasurement2	3.0
IMeasurement3	4.0
IMeasurement4	4.2
IMeasurement5	5.0
IMeasurement7	6.2
IMeasurement8	7.2
IMeasurement9	8.35
IMeasurement10	8.35
IMeasurement11	8.50
IMeasurement12	9.00
IMeasurement13	9.20
IMeasurement14	9.22
IMeasurement15	9.40

IArrayTransfer Interface

Description

Contains methods for putting data in and getting data out of the analyzer using typed data. This interface transfers data more efficiently than the IMeasurement Interface. However, this interfaces is

only usable from VB6, C, & C++.

Methods	Description
getComplex	Retrieves real and imaginary data from the specified buffer.
getNAComplex	Retrieves typed NAComplex data from the specified buffer.
getPairedData	Retrieves magnitude and phase data pairs from the specified buffer.
getScalar	Retrieves scalar data from the specified buffer.
putComplex	Puts real and imaginary data into the specified buffer.
putNAComplex	Puts typed NAComplex data into the specified buffer.
putScalar	Puts scalar data into the measurement result buffer.

Properties	Description
------------	-------------

None

IArrayTransfer History

Interface	Introduced with PNA Rev:
IArrayTransfer	1.0

MeasurementClassProperties Object

Description

These properties return properties of specific measurement classes.

Accessing the MeasurementClassProperties object

```
Set app = CreateObject("AgilentPNA835x.Application")  
Set cap = app.Capabilities  
  
'Access the MeasurmentClassProperties Object  
  
Set measProps = cap.MeasurmentClassProperties ("Swept IMD")  
  
list=measProps.SupportedParameters
```

See Also

- [MeasurementClassProperties Property](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods

Interface

[See History](#)

None

Properties

Description

SupportedGenericParameters MeasurementClassProperties Not supported at this time.

[SupportedParameters](#) MeasurementClassProperties

IMeasurementClassProperties History

	Interface	Introduced with PNA Rev:
IMeasurementClassProperties		9.40

Last Modified:

24-May-2011 MX New topic

Measurement Collection

Description

A collection object that provides a mechanism for iterating through the Application measurements.

Accessing the Measurements collection

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim measments As Measurements
Set measments = app.Measurements
```

See Also:

- [Measurement Object](#)
- [Collections in the Analyzer](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods	Description
Add	Adds a Measurement to the collection.
Item	Use to get a handle on a measurement in the collection.
Remove	Removes a measurement from the measurements collection.

Properties	Description
Count	Returns the number of measurements in the analyzer.
Parent	Returns a handle to the current Application.

NAWindow Object

Description

The NAWindow object controls the part of the display that contains the graticule, or what is written on the display.

Accessing the NaWindow object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim window As NAWindow
Set window = app.NAWindows(1)
window.AutoScale
```

OR

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", "analyzerName")

app.NAWindows(1).AutoScale
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)

(**Bold** Methods or Properties provide access to a child object)

Methods	Description
Autoscale	Autoscales all measurements in the window. Shared with the Trace Object
ShowMarkerReadout	Shows and Hides the Marker readout for the active marker in the upper-right corner of the window object.
ShowTable	Shows or Hides the specified table for the active measurement in the lower part of the window object.
Properties	Description

<u>ActiveTrace</u>	Sets a trace to the Active Trace.
<u>MarkerReadout</u>	Sets and reads the state of the Marker readouts.
<u>MarkerReadoutResponsePlaces</u>	
<u>MarkerReadoutSize</u>	Specifies the size of font used when displaying Marker readout in the selected window.
<u>MarkerReadoutsPerTrace</u>	
<u>MarkerReadoutStimulusPlaces</u>	
<u>MarkerReadoutXPosition</u>	
<u>MarkerReadoutYPosition</u>	
<u>MarkerSymbol</u>	
<u>OneMarkerReadoutPerTrace</u>	Superseded with <u>MarkerReadoutsPerTrace Property</u>
<u>ScaleCouplingMethod</u>	Sets and returns the method of scale coupling.
<u>ScaleCouplingState</u>	Enables and disables scale coupling for the specified window.
<u>Title</u>	Writes or reads a custom title for the window.
<u>TitleState</u>	Turns ON and OFF the window title.
<u>Traces</u>	Collection for getting a handle to a trace or iterating through the traces in a window.
<u>WindowNumber</u>	Reads the number of the active window.
<u>WindowState</u>	Maximizes or minimizes a window.
	Shared with the Application Object

INaWindow History

Interface	Introduced with PNA Rev:
INaWindow	1.0
INaWindow2	9.0

NAWindows Collection

Description

A collection object that provides a mechanism for iterating through the Application windows.

Accessing the NaWindows collection

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim windows As NAWindows
Set windows = app.NAWindows
```

See Also:

- [NAWindow Object](#)
- [Collections in the Analyzer](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods	Description
Add	Adds a window to the NAWindows collection.
Item	Use to get a handle to a window in the collection.
Remove	Removes a window from the NAWindows collection.

Properties	Description
Count	Returns the number of windows on the analyzer.
Parent	Returns a handle to the current Application.

NoiseCal Object

Description

Controls the noise figure calibration settings for amplifiers and converters. These commands supplement the standard calibration commands on the [GuidedCalibration Object](#).

Accessing the NoiseCal object

```
Dim app as AgilentPNA835x.Application

Set noiseCal = pna.GetCalmanager.CreateCustomCalEx\(channelNum\)

Set noiseCalExtension = noiseCal.CustomCalConfiguration

noiseCalExtension.NoiseSourceCold = 300
```

See Also

- **Examples:**
 - [Create and Cal a Noise Figure Measurement](#)
 - [Create and Cal an NFX Measurement](#)
- [NoiseFigure Object](#)
- [About Noise Figure Measurements](#)
- [About NFX Measurements](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)

Method	Interface	Description
	See History	

None

Property	Interface	Description
AutoOrientTuner	INoiseCal2	Sets the state of auto orientation for a noise tuner during Noise Figure for NFX.
CalMethod	INoiseCal	Sets and returns the method for performing calibration on a noise channel.
EnableLOPowerCal	INoiseCal2	Enables and disables LO power calibration for NFX.
ENRFile	INoiseCal	Sets and returns the name of the ENR file associated with the noise source.
ForceDeEmbedENRAdapter	INoiseCal2	Sets and reads the state of ENR adapter de-embedding.
ForceDeEmbedSensorAdapter	INoiseCal2	Sets and reads the state of noise source adapter de-embedding.
NoiseSourceCalKitType	INoiseCal	Sets and reads the Cal Kit type used to perform a cal at the adapter which is used to connect the noise source (if required.)
NoiseSourceCold	INoiseCal	Sets and returns the current temperature at the noise source.
NoiseSourceConnectorType	INoiseCal	Sets and reads the connector type of the noise source used during the cal.
RcvCharMethod	INoiseCal3	Set and read the method used to characterize the noise receivers.

NoiseConfiguration History

Interface	Introduced with PNA Rev:
INoiseCal	8.0
INoiseCal2	9.10
INoiseCal3	9.70

Last Modified:

9-Jun-2011	Added INoise3
30-May-2007	MX New topic

NoiseFigure Object

Description

Controls the Noise Figure application settings.

Accessing the NoiseFigure object

```
Dim app as AgilentPNA835x.Application  
app.CreateCustomMeasurementEx(1, "NoiseFigure", "NF", 1)  
  
Dim NoiseFig  
  
Set NoiseFig = app.ActiveChannel.CustomChannelConfiguration
```

See Also:

- Example programs
 - [Create and Cal a NoiseFigure Measurement](#)
 - [Create and Cal an NFX Measurement](#)
- [About Noise Figure Measurements](#)
- [Noise Figure Calibration Object](#)
- [app.NoiseSourceState \(ON and OFF\)](#)
- [ENRFile Object](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)

Method	Interface	Description
	See History	
GetSnPData	INoiseFigure7	Reads noise parameter snp data.
SetPortMap	INoiseFigure6	Maps DUT ports to PNA-X ports (Opt 028)
WriteSnPData	INoiseFigure7	Reads noise parameter data to snp file.

Property	Interface	Description
AmbientTemperature	INoiseFigure	Sets the air temperature at which the measurement is being performed.
DeviceInputPort	INoiseFigure6	Read DUT input port map
DeviceOutputPort	INoiseFigure6	Read DUT output port map
ImpedanceStates	INoiseFigure	Sets the number of impedance states to use during calibrated measurements.
NoiseAverageFactor	INoiseFigure	Set averaging of noise receiver.
NoiseAverageState	INoiseFigure	Turn noise averaging ON and OFF
NoiseBandwidth	INoiseFigure	Set bandwidth of noise receiver.
NoiseGain	INoiseFigure	Set gain state of noise receiver.
NoiseReceiver	INoiseFigure5	Sets and returns the receiver to use for noise measurements.
NoiseReceiverSweepTime	INoiseFigure3	Returns an estimate of sweep time.
NoiseTuner	INoiseFigure	Sets and returns the noise tuner identifier,
NoiseTunerIn	INoiseFigure	Sets and returns the port identifier of the ECal noise tuner Input
NoiseTunerOut	INoiseFigure	Sets and returns the port identifier of the ECal noise tuner Output
SourcePullForSParameters	INoiseFigure4	Set and read the use of source pull technique to compute S22 on Noise Figure on Converters.

NoiseFigure History

Interface	Introduced with PNA Rev:
INoiseFigure	8.0
INoiseFigure3	9.0
INoiseFigure4	9.1
INoiseFigure5	9.2
INoiseFigure6	9.22
INoiseFigure7	9.80

Last Modified:

- 24-Oct-2012 Added NF7
- 10-May-2012 Fixed typo
- 30-Apr-2010 Added NF6
- 29-May-2007 MN New topic

PathConfiguration Object

Description

Provides access to the path configuration currently active on the channel object.

To load, store, or delete a configuration, see [ConfigurationManager](#) Object.

Accessing the PathConfiguration object in VB

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim chan as Channel
Set chan = app.ActiveChannel

Dim pathConfig As PathConfiguration
Set pathConfig = chan.PathConfiguration
```

Accessing the PathConfiguration object in C#

```
Type pnaType = Type.GetTypeFromProgID("AgilentPNA835x.Application", "PNA-NAME-HERE");

AgilentPNA835x.Application pna =
(AgilentPNA835x.Application)Activator.CreateInstance(pnaType);

AgilentPNA835x.Channel chan = (AgilentPNA835x.Channel)pna.ActiveChannel;

AgilentPNA835x.PathConfiguration path =
(AgilentPNA835x.PathConfiguration)chan.get_PathConfiguration();
```

Note:

To learn how to make configuration (element) settings, see this [Path Configuration Example](#)

Also see this [list of configurable elements and settings](#).

See Also:

- [PathConfigurationManager Object](#)
- [PathElement Object](#)
- [Path Configurator](#) UI
- [PNA Automation Interfaces](#)

- [The PNA Object Model](#)

Methods	Interface	Description
	See History	

CopyFrom	IPathConfiguration2	Copy the mechanical switch settings and attenuator settings from the specified channel to the active channel.
Store	IPathConfiguration	Saves the current configuration to the specified name.

Properties	Interface	Description
	See History	

DescriptiveText	IPathConfiguration	Write and read descriptive text associated with the configuration.
Elements	IPathConfiguration	Collection of Elements that can be configured (switches and so forth). See the list of elements and settings.
Element	IPathConfiguration	Returns a handle to a IPathElement object.
Name	IPathConfiguration	Returns the name of the current configuration.
Parent	IPathConfiguration	Returns a pointer to the parent COM object (Channel).

IPathConfiguration History

Interface	Introduced with PNA Rev:
IPathConfiguration	7.2
IPathConfiguration2	9.4

PathConfigurationManager Object

Description

These commands allow configurations to be stored, loaded, or deleted on the PNA.

To **make** path configuration settings, see [PathConfiguration Object](#) and the [PathElement Object](#)

Accessing the PathConfigurationManager object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim pathConfig As PathConfigurationManager
Set pathConfig = app.PathConfigurationManager
```

Note:

To learn how to make configuration (element) settings, see this [Path Configuration Example](#)

Also see this [list of configurable elements and settings](#).

See Also:

- [Path Configuration Example](#)
- [Path Configurator](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods	Interface	Description
---------	-----------	-------------

[See History](#)

DeleteConfiguration	IPathConfigurationManager	Deletes the specified configuration from the PNA.
-------------------------------------	---------------------------	---

LoadConfiguration	IPathConfigurationManager	Loads the named configuration.
-----------------------------------	---------------------------	--------------------------------

StoreConfiguration	IPathConfigurationManager	Saves the path configuration
------------------------------------	---------------------------	------------------------------

Properties	Interface	Description
------------	-----------	-------------

[See History](#)

Configurations	IPathConfigurationManager	Returns a list of configuration names stored in the PNA.
--------------------------------	---------------------------	--

Parent	IPathConfigurationManager	Returns a handle to the Application object.
------------------------	---------------------------	---

IPathConfigurationManager History

Interface	Introduced with PNA Rev:
-----------	--------------------------

IPathConfigurationManager	7.2
---------------------------	-----

PathElement Object

Description

Provides access to the settings for the PathElement object.

Accessing the PathElement object in VB

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim chan as Channel
Set chan = app.ActiveChannel

Dim pathConfig As PathConfiguration
Set pathConfig = chan.PathConfiguration

Dim element as PathElement
Set element = pathConfig.PathElement("Src1")
```

Accessing the PathElement object in C#

```
Type pnaType = Type.GetTypeFromProgID("AgilentPNA835x.Application", "PNA-NAME-HERE");

AgilentPNA835x.Application pna =
(AgilentPNA835x.Application)Activator.CreateInstance(pnaType);

AgilentPNA835x.Channel chan = (AgilentPNA835x.Channel)pna.ActiveChannel;

AgilentPNA835x.PathConfiguration path =
(AgilentPNA835x.PathConfiguration)chan.get_PathConfiguration();

path.get_Element("Port1RefMxr").Value = "External";
```

Note:

To learn how to make configuration (element) settings, see this [Path Configuration Example](#)

Also see this [list of configurable elements and settings](#).

See Also:

- [Path Configurator](#)
- [PathConfigurationManager Object](#)
- [PathConfiguration Object](#)

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods	Interface	Description
	See History	

None

Properties	Interface	Description
	See History	

Name	IPathElement	Returns the name of the element.
Parent	IPathElement	Returns a pointer to the Parent Object (PathConfiguration)
Value	IPathElement	Read / Write get the current setting for the element.
Values	IPathElement	Returns all valid settings for the element.

IPathElement History

Interface	Introduced with PNA Rev:
IPathElement	7.2

PhaseControl Object

Description

Contains the properties for configuring Phase Sweep (Opt 088) in the PNA.

Accessing the PhaseControl object

```
Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel
Dim phase as PhaseControl
Set phase = chan.PhaseControl
```

See Also:

- [About Phase Control](#)
- Set Phase Sweep using [SweepType Property](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Program](#)

Methods	Description
---------	-------------

None

Properties	Interface	Description
------------	-----------	-------------

[See History](#)

CouplePhasePortSettings	IPhaseControl	Set and return whether to couple phase control settings.
FixedPhase	IPhaseControl	Set and return the fixed phase value.
FixedRatioedPower	IPhaseControl	Set and return the fixed ratioed power value.
PhaseControlMode	IPhaseControl	Set and return the ON/Off state of phase control.
PhaseCorrectionData	IPhaseControl	Set and return an array of phase offsets.
PhaseCorrectionEnabled	IPhaseControl	Set and return whether to use the phase correction offset array.

<u>PhaseIterationNumber</u>	IPhaseControl	Set and return max number of leveling sweeps
<u>PhaseParameter</u>	IPhaseControl	Set and return the ratioed receivers (parameter) to use for phase control.
<u>PhaseParameterModes</u>	IPhaseControl	Returns the available phase control modes for the specified port.
<u>PhaseReferencePort</u>	IPhaseControl	Sets and returns the reference port for the Phase Control measurement.
<u>PhaseTolerance</u>	IPhaseControl	Set and return tolerance value for leveling sweeps
<u>RatioedPowerCorrectionData</u>	IPhaseControl	Set and return ratioed power offset data
<u>RatioedPowerCorrectionEnabled</u>	IPhaseControl	Set and return whether to use the ratioed power offset array
<u>StartPhase</u>	IPhaseControl	Set and return the start value of phase sweep.
<u>StartRatioedPower</u>	IPhaseControl	Set and return the start ratioed power value.
<u>StopPhase</u>	IPhaseControl	Set and return the stop value of phase sweep.
<u>StopRatioedPower</u>	IPhaseControl	Set and return the stop ratioed power value.

IPhaseControl History

Interface	Introduced with PNA Rev:
IPhaseControl	9.33

Last Modified:

3-Dec-2010 MX New topic

PhaseReferenceCalibration Object

Description

Use this interface to perform a Phase Reference Calibration. This calibration is performed as a 'Tier 1' cal. After it is performed and saved, it can then be recalled into an SMC+Phase Calibration.

- It is NOT necessary to create an SMC measurement before performing a **remote** Phase Reference Cal. It is necessary when performed from the user interface.
- Port selection is made remotely by selecting connectors and Cal Kits for the ports to be included in the SOLT calibration.

Accessing the PhaseReferenceCalibration Object

```
Dim app as AgilentPNA835x.Application

Dim CalMgr

Set CalMgr = app.GetCalManager

Dim phaseRefCal

Set phaseRefCal = CalMgr.PhaseReferenceCalibration

'Then get a handle to the GuidedCal object

Dim guidedCal

Set guidedCal = phaseRefCal.GuidedCalibration
```

See Also:

- Example program
- Learn about Calibrate All Channels .
- PNA Automation Interfaces
- The PNA Object Model
- **Superseded** commands

(**Bold** Methods or Properties provide access to a child object)

In the following table IPhaseReferenceCalibration is abbreviated to IPhaseRefCal

Methods	Interface	Description
	See History	

GetConnectedPhaseReferences	IPhaseRefCal	Reads the ID strings of the phase references that are currently connected to the PNA USB
Reset	IPhaseRefCal	Resets all properties associated with the Cal All session to their default values.

Properties

CalKitType	IPhaseRefCal2	Sets and returns the Cal Kit to be used during the S-parameter portion of a Phase Reference calibration.
CalSet	IPhaseRefCal	Sets and reads the Cal Set name into which the calibration will be saved.
GuidedCalibration	IPhaseRefCal	Provides access to the GuidedCal object. Use this to perform the Calibration.
IncludePort	IPhaseRefCal2	Sets and returns the enable state for the specified port.
IncludeUnknownMixer	IPhaseRefCal2	Sets and returns the state of Unknown Mixer calibration.
PhaseReference	IPhaseRefCal	Sets and returns the Phase Reference ID to be used for the Phase Reference calibration.
SourceAttenuator	IPhaseRefCal	Sets and returns the Source Attenuator setting.
StartFrequency	IPhaseRefCal	Sets and returns the phase reference cal start frequency.
StopFrequency	IPhaseRefCal	Sets and returns the phase reference cal stop frequency.
UnknownMixerInputPower	IPhaseRefCal2	Sets and returns the input power level to the unknown mixer.

UnknownMixerLOFrequency IPhaseRefCal2 Sets and returns the LO Frequency of the unknown mixer.

UnknownMixerLOPower IPhaseRefCal2 Sets and returns the LO power level to the unknown mixer.

IPhaseReferenceCalibration History

Interface	Introduced with PNA Rev:
IPhaseReferenceCalibration	9.70

Last modified:

11-Mar-2012 New topic

PNOP Object

Description

Contains the methods and properties that initiate and return Power Normal Operating Point markers.

Accessing the PNOP Object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application")

Dim pnop As PNOP
Set pnop = app.ActiveMeasurement.PNOP
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [PNOP Markers](#)
- [Example Programs](#)

Methods	Interface	Description
SearchPowerNormalOperatingPoint	IPNOP	Initiates a PNOP search
Properties		Description
BackOff	IPNOP	Sets and returns the backoff value.
BackOffGain	IPNOP	Returns the BackOffGain result.
BackOffPIn	IPNOP	Returns the BackOffPIn result
BackOffPout	IPNOP	Returns the BackOffPout result
Compression	IPNOP	Returns the Compression result
CompressionMax	IPNOP	Returns the Compression Max result
Gain	IPNOP	Returns the Gain result
GainMax	IPNOP	Returns the Gain Max result
Pin	IPNOP	Returns the Pin result
PinOffset	IPNOP	Sets and returns the PinOffset value
PMaxIn	IPNOP	Returns the PMaxIn result

[PMaxOut](#)

IPNOP Returns the PMaxOut result

[POut](#)

IPNOP Returns the P Out result

IPNOP History

Interface	Introduced with PNA Rev:
IPNOP	A.09.20

Last Modified:

19-Feb-2010 MX New topic

PortExtension Object **Superseded**

ALL methods and properties on the PortExtension Object are Superseded with the [Fixturing Object](#).

Description

Contains the methods and properties that control Port Extensions.

Accessing a PortExtension object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim PortExt As PortExtension
Set PortExt = app.PortExtension
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)
- [Superseded commands](#)

Methods

None

Property	Description
Input A	Sets the Input A extension value.
Input B	Sets the Input B extension value.
Input C	Sets the Input C extension value.
Port 1	Sets the Port 1 extension value.
Port 2	Sets the Port 2 extension value.
Port 3	Sets the Port 3 extension value.
State	Turns Port Extensions ON and OFF.

IPort Extension History

Interface	Introduced with PNA Rev:
IPort Extension	1.0

PowerLossSegment Object

Description

Contains the properties describing a segment of the power loss table used in source power calibration.

You can get a handle to one of these segments through the [segments.Item](#) Method of the PowerLossSegments collection.

Accessing the PowerLossSegment object

You can get a handle to one of these segments through `PowerLossSegments.Item(n)`

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim PwrLossSeg As PowerLossSegment
Set PwrLossSeg = app.SourcePowerCalibrator.PowerLossSegments(1)
```

See Also:

- [About Source Power Cal](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods

None

Properties	Description
Frequency	The frequency (Hz) associated with this segment.
Loss	The loss value (dB) associated with this segment.
SegmentNumber	Returns the number of this segment

IPowerLossSegment History

Interface

**Introduced with PNA
Rev:**

IPowerLossSegment

2.0

PowerLossSegments Collection

Description

A collection object that provides a mechanism for iterating through the segments of the power loss table used in source power calibration. The power loss table can contain up to 9999 segments.

Accessing the PowerLossSegments collection

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim PwrLossSegs As PowerLossSegments
Set PwrLossSegs = app.SourcePowerCalibrator.PowerLossSegments
```

See Also:

- [PowerLossSegment Object](#)
- [Collections in the Analyzer](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods	Description
Add	Adds a PowerLossSegment object to the collection.
Item	Use to get a handle to a PowerLossSegment object in the collection.
Remove	Removes an object from the collection.

Properties	Description
Count	Returns the number of objects in the collection.
Parent	Returns a handle to the Parent object (SourcePowerCalibrator) of this collection.

Last Modified:

14-May-2012	Increase segment limit
2-Jun-2008	Added segment limit

PowerLossSegmentPMAR Object

Description

Contains the properties describing a segment of the power loss table used with a Power Meter as Receiver.

Accessing the PowerLossSegmentPMAR object

You can get a handle to one of these segments through [PowerLossSegmentsPMAR.Item\(n\)](#)

See Also

- **Example:** [Create a PMAR Device and Measurement](#)
- [Configure an External Device](#)
- [About PMAR](#)
- [ExternalDevice Object](#)
- [The PNA Object Model](#)

Methods

None

Properties

Description

Frequency	The frequency (Hz) associated with this segment.
Loss	The loss value (dB) associated with this segment.
SegmentNumber	Returns the number of this segment

IPowerLossSegment History

Interface	Introduced with PNA Rev:
IPowerLossSegmentPMAR	9.0

Last Modified:

25-Aug-2009 MX New topic

PowerLossSegmentsPMAR Collection

Description

A collection object that provides a mechanism for iterating through the segments of the power loss table used with a Power Meter as Receiver. The power loss table can contain up to 9999 segments.

Accessing the PowerLossSegmentsPMAR collection

- **Example:** [Create a PMAR Device and Measurement](#)

See Also

- [Configure an External Device](#)
- [About PMAR](#)
- [ExternalDevice Object](#)
- [The PNA Object Model](#)

Methods	Description
Add	Adds a PowerLossSegmentPMAR object to the collection.
Item	Use to get a handle to a PowerLossSegmentPMAR object in the collection.
Remove	Removes a PowerLossSegmentPMAR object from the collection.
Properties	Description
Count	Returns the number of PowerLossSegmentPMAR objects in the collection.
Parent	Returns a handle to the Parent object of this collection.

Last Modified:

14-May-2012	Increase segment limit
25-Aug-2009	MX New topic

PowerMeterInterface Object

Description

Contains the properties used to select a power meter and sensor to be used for a source power calibration.

Note: This object replaces the [PowerMeterGPIBAddress Property](#).

Accessing the PowerMeterInterface object

Get a handle to a power meter object using the [PowerMeterInterfaces](#) collection.

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)
Dim pwrMtrInterfaces As PowerMeterInterfaces
Set pwrMtrInterfaces = app.SourcePowerCalibrator.PowerMeterInterfaces
If pwrMtrInterfaces.Count > 0 Then
    Dim pwrMtrInterface As PowerMeterInterface
    Set pwrMtrInterface = pwrMtrInterfaces(1)
    pwrMtrInterface.Path = naUSB
    pwrMtrInterface.Locator = "Agilent Technologies,U2000A,MY12345678"
End If
```

See Also:

- [Source Power Calibration](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods

None

Properties

Description

[Path](#)

Specifies the interface to use: GPIB, USB, LAN

[Locator](#)

Specifies the location (address) of the power meter/sensor.

IPowerMeterInterface History

Interface	Introduced with PNA Rev:
IPowerMeterInterface	7.50

Last Modified:

5-Jul-2007 MX New topic

PowerMeterInterfaces Collection

Description

A collection object that provides a mechanism for accessing the PowerMeterInterface objects.

The collection size is limited to one PowerMeterInterface object. By default, that PowerMeterInterface object refers to GPIB, and to the GPIB address that is currently set for the power meter on that PNA.

The power meter is specified by using the [Path](#) property.

Accessing the PowerMeterInterfaces collection

Get a handle to a power meter object using the PowerMeterInterfaces collection.

```
Dim app As AgilentPNA835x.Application

Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim pwrMtrInterfaces As PowerMeterInterfaces

Set pwrMtrInterfaces = app.SourcePowerCalibrator.PowerMeterInterfaces

If pwrMtrInterfaces.Count > 0 Then

    Dim pwrMtrInterface As PowerMeterInterface

    Set pwrMtrInterface = pwrMtrInterfaces(1)

    pwrMtrInterface.Path = naUSB

    pwrMtrInterface.Locator = "Agilent Technologies,U2000A,MY12345678"

End If
```

See Also:

- [Source Power Calibration](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods

[Item](#) Use to get a handle to a [PowerMeterInterface](#) object in the collection.

Properties

Properties	Description
------------	-------------

Count	Returns the number of objects in the collection.
-----------------------	--

Parent	Returns a handle to the SourcePowerCalibrator
------------------------	---

IPowerMeterInterfaces History

Interface	Introduced with PNA Rev:
IPowerMeterInterfaces	7.50

Last Modified:

5-Mar-2009	Added Parent Property
9-Jul-2007	MX New topic

PowerSensor Object

Description

Each power sensor connected to the power meter associated with Source Power Calibration will have a PowerSensor object created to represent it. These PowerSensor objects reside in the [PowerSensors](#) collection within the SourcePowerCalibrator object. You cannot directly create PowerSensor objects, but can only retrieve existing ones from the PowerSensors collection.

The PowerSensorCalFactorSegment object is also accessed through the PowerSensor object. These are accessed through the CalFactorSegments collection in the PowerSensor object.

Accessing a PowerSensor object

```
Dim pna As AgilentPNA835x.Application
Set pna = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim powerCalibrator as SourcePowerCalibrator
Dim powerSensor as PowerSensor
Dim calFactorSegment as PowerSensorCalFactorSegment

Set powerCalibrator = pna.SourcePowerCalibrator

' Specify GPIB address of the power meter.
powerCalibrator.PowerMeterGPIBAddress = 13

' Each time the PowerSensors collection is accessed, the power meter is queried to
determine which channels have sensors attached. The collection is updated
accordingly.

If powerCalibrator.PowerSensors.Count > 0 Then
' If channel B of the meter has a sensor attached but channel A does not, then
element 1 of the
' collection is sensor B. Whenever channel A has a sensor, sensor A will be element
1.
Set powerSensor = powerCalibrator.PowerSensors(1)
' Insert one new PowerSensorCalFactorSegment at the beginning of the collection
(index 1).

powerSensor.CalFactorSegments.Add(1)
' Assign our variable to refer to that object.
Set calFactorSegment = powerSensor.CalFactorSegments(1)

' Set property values for that object.
calFactorSegment.Frequency = 300000
' frequency in Hz
calFactorSegment.CalFactor = 98
' cal factor in percent
```

End If

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)

(**Bold** Methods or Properties provide access to a child object)

Methods	Description
---------	-------------

None

Properties	Description
------------	-------------

CalFactorSegments	Collection for iterating through the segments of a power sensor cal factor table.
MaximumFrequency	Maximum usable frequency (Hz) specified for this power sensor.
MinimumFrequency	Minimum usable frequency (Hz) specified for this power sensor.
PowerMeterChannel	Identifies which power sensor this object corresponds to (or which channel of the power meter the sensor is connected to).
ReferenceCalFactor	Reference cal factor (%) associated with this power sensor.

IPowerSensor History

Interface	Introduced with PNA Rev:
IPowerSensor	2.0

PowerSensors Collection

Description

A collection object that provides a mechanism for iterating through the PowerSensor objects which are connected to the power meter. Each time this collection object is accessed, the power meter is queried to determine how many sensors are connected to it. The collection size and order of objects is then adjusted accordingly before the requested method or property operation is performed. The power meter is specified by using the [PowerMeterGPIBAddress](#) property of the [SourcePowerCalibrator](#) object.

Accessing the PowerSensors Collection

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim PwrSensors As PowerSensors
Set PwrSensors = app.SourcePowerCalibrator.PowerSensors
```

See Also:

- [PowerSensor Object](#)
- [Collections in the Analyzer](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods	Description
Item	Use to get a handle to a PowerSensor object in the collection.
Properties	Description
Count	Returns the number of objects in the collection.
Parent	Returns a handle to the Parent object (SourcePowerCalibrator) of this collection.

PowerSensorCalFactorSegment Object

Description

Contains the properties describing a segment of a power sensor cal factor table.

Accessing the PowerSensorCalFactorSegment object

You can get a handle to one of these segments through [CalFactorSegments.Item\(n\)](#)

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim calFactSeg As CalFactorSegments
Set calFactSeg = app.SourcePowerCalibrator.PowerSensors(1).CalFactorSegments(1)
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods

None

Properties

Description

[Frequency](#)

The frequency (Hz) associated with this segment.

Shared with the PowerLossSegment Object

[CalFactor](#)

The cal factor (%) associated with this segment.

[SegmentNumber](#)

Returns the number of this segment.

Shared with the PowerLossSegment Object

IPowerSensorCalFactorSegment History

Interface	Introduced with PNA Rev:
------------------	---

IPowerSensorCalFactorSegment	2.0
------------------------------	-----

PowerSensorCalFactorSegmentPMAR Object

Description

Contains the properties describing a segment of a power sensor cal factor table that is used with Power Meter as Receiver.

Accessing the PowerSensorCalFactorSegment PMAR object

You can get a handle to one of these segments through [CalFactorSegmentsPMAR.Item\(n\)](#).

See Also

- **Example:** [Create a PMAR Device and Measurement](#)
- [Configure an External Device](#)
- [ExternalDevice Object](#)
- [The PNA Object Model](#)

Methods

None

Properties

Description

[Frequency](#)

The frequency (Hz) associated with this segment.

[CalFactor](#)

The cal factor (%) associated with this segment.

[SegmentNumber](#)

Returns the number of this segment.

IPowerSensorCalFactorSegmentPMAR History

	Interface	Introduced with PNA Rev:
	IPowerSensorCalFactorSegmentPMAR	9.0

Last Modified:

25-Aug-2009 MX New topic

PowerSensorAsReceiver Object

Description

Provides the settings for configuring a Power Meter to be used as a PNA Receiver (PMAR).

Accessing a PowerSensorAsReceiver object

This object is accessed through ExternalDevice.ExtendedProperties. When an external device is added to the [ExternalDevices collection](#), and the DeviceType property is set to Power Meter, then ExtendedProperties is used to get a handle to this object.

```
externalDevices.Add "NewPMAR"  
  
dim newExternalDevice  
  
Set newExternalDevice = externalDevices.Item("NewPMAR")  
  
newExternalDevice.DeviceType = "Power Meter"  
  
dim PMAR  
  
Set PMAR = newExternalDevice.ExtendedProperties  
  
PMAR.ReadingsPerPoint = 10
```

See Also:

- **Example:** [Create a PMAR Device and Measurement](#)
- [Configure an External Device](#)
- [ExternalDevice Object](#)
- [The PNA Object Model](#)

(**Bold** Methods or Properties provide access to a child object)

Methods**Description**

None

Properties**Description****[CalFactorSegmentsPMAR](#)**

Provides access to the collection of segments of a power sensor cal factor table.

[LimitFrequency](#)

Enable or disable the use of the power meter min and max frequencies.

[MaximumFrequency](#)

Maximum usable frequency (Hz) specified for this power sensor.

[MinimumFrequency](#)

Minimum usable frequency (Hz) specified for this power sensor.

[PowerLossSegmentsPMAR](#)

Provides access to the collection of segments of the power loss table.

[PowerMeterChannel](#)

Identifies which power sensor this object corresponds to (or which channel of the power meter the sensor is connected to).

[ReadingsPerPoint](#)

Allows for settling of the power sensor READINGS.

[ReadingsTolerance](#)

Allows for settling of the power sensor READINGS.

[ReferenceCalFactor](#)

Reference cal factor (%) associated with this power sensor.

[SensorIndex](#)

Sets the power sensor channel (1 or 2) to be used.

[UsePowerLossSegments](#)

Specifies if subsequent power readings will use of the loss table.

IPowerSensorAsReceiver History

Interface	Introduced with PNA Rev:
IPowerSensorAsReceiver	9.0

Last Modified:

27-Aug-2009 MX New topic

Preferences Object

Description

Sets the preferences for the behavior of several properties.

Accessing the Preferences object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim pref As Preferences
Set pref = app.Preferences
```

See Also:

- [PNA Preferences](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)

(**Bold** Methods or Properties provide access to a child object)

Methods

RestoreDefaults	IPreferences9	Restores preference settings to their factory defaults.
---------------------------------	---------------	---

Properties

Interface

Description

[See History](#)

AuxTriggerScopeIsGlobal	IPreferences5	Sets the External Trigger OUT behavior to have either Global or Channel scope.
---	---------------	--

CitiContents	IPreferences	Specifies the contents of subsequent citifile saves. Superseded with SaveData
------------------------------	--------------	--

CitiFormat	IPreferences	Specifies the format of subsequent citifile saves. Superseded with SaveData
----------------------------	--------------	--

<u>DisplayColors</u>	IPreferences10	Provides access to the <u>ComColors</u> Object.
<u>EnableSourceUnleveledEvents</u>	IPreferences6	Specifies whether or not to report Source Unleveled errors as system events.
<u>ExternalDeviceDeActivatePolicy</u>	IPreferences10	External Devices remain activated or are de-activated when the PNA is Preset or when a Instrument State is recalled.
<u>FrequencyOffsetRangeForCalComputations</u>	IPreferences10	Specifies the FOM frequency range to use when performing calibration.
<u>IMDECalExtrapolation</u>	IPreferences14	Allow ECal beyond stop frequency for IMD apps.
<u>OffsetReceiverAttenuator</u>	IPreferences6	Mathematically offset the test port receiver.
<u>OffsetSourceAttenuator</u>	IPreferences6	Mathematically offset the reference receiver.
<u>Port1NoiseTunerSwitchPresetsToExternal</u>	IPreferences8	Sets default setting for Noise Figure switch.
<u>PowerOnDuringRetraceMode</u>	IPreferences4	Specify whether to turn RF power ON or OFF during a retrace for single-band frequency or segment sweeps ONLY.
<u>PowerSweepRetracePowerMode</u>	IPreferences3	At the end of a power sweep, specifies whether to maintain source power at the start or stop power level.
<u>PreferInternalTriggerOnChannelSingle</u>	IPreferences2	Sets the preference for chan.Single behavior.
<u>PreferInternalTriggerOnUnguidedCal</u>	IPreferences2	Set the preference for the trigger behavior when performing an Unguided calibration.
<u>PresetPowerState</u>	IPreferences11	Set and return the Preset Power ON/OFF state.
<u>PrintColors</u>	IPreferences10	Provides access to the <u>ComColors</u> Object.
<u>RecallSoftkeysMostRecent</u>	IPreferences13	List files for recall on softkeys by name or alphabetically.

[RemoteCalStoragePreference](#)

IPreferences7 Specifies the default manner in which calibrations performed via SCPI or COM are to be stored.

[ReportReceiverOverload](#)

IPreferences12 Set whether to display receiver overload warnings.

[RFOffOnReceiverOverload](#)

IPreferences12 Set whether to turn source power OFF when a receiver is overloaded.

[SnPFormat](#)

IPreferences Specifies the format of subsequent .S1P, .S2P, .S3P file saves.

[TwoPointGroupDelayAperture](#)

IPreferences11 Sets Group Delay Aperture default to 2 points.

IPreferences History

Interface	Introduced with PNA Rev:
IPreferences	4.0
IPreferences2	6.0
IPreferences3	7.2
IPreferences4	6.04
IPreferences5	7.10
IPreferences6	7.20
IPreferences7	7.21
IPreferences8	8.0
IPreferences9	8.2
IPreferences10	9.0
IPreferences11	9.20
IPreferences12	9.30
IPreferences13	9.50
IPreferences14	10.0

PSaturation Object

Description

Contains the methods and properties that initiate a Power Saturation marker search and returns PSAT data.

Accessing the PSaturation Object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application")

Dim PSat As PSaturation
Set PSat = app.ActiveMeasurement.PSaturation
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [PSaturation Markers](#)
- [Example Programs](#)

Methods	Interface	Description
SearchPowerSaturation	IPSaturation	Initiates a Power Saturation marker search.
Properties		Description
CompressionMax	IPSaturation	Returns the Compression Max result of a PSat marker search.
CompressionSaturation	IPSaturation	Returns the Compression Saturation result of a PSat marker search.
GainLinear	IPSaturation	Returns the Linear Gain result of a PSat marker search.
GainMax	IPSaturation	Returns the GainMax result of a PSAT marker search.
GainSaturation	IPSaturation	Returns the GainSaturation result of a PSAT marker search.
Pin	IPSaturation	Returns the Pin result of a PSAT marker search.
PMaxBackOff	IPSaturation	Sets and returns the backoff value used to calculate various PSAT parameters.
PMaxIn	IPSaturation	Returns the PMaxIn result of a PSAT marker search.

[PMaxOut](#)

IPSaturation Returns the PMaxOut result of a PSAT marker search.

[POut Property](#)

IPSaturation Returns the POut result of a PSAT marker search.

IPSaturation History

Interface	Introduced with PNA Rev:
IPSaturation	A.09.20

PulseGenerator Object

Description

Contains the properties for configuring the five internal pulse generators in the PNA-X.

[Learn more about the PNA-X Pulse Generators.](#)

Accessing the PulseGenerator object

```
Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel
Dim pulse as PulseGenerator
Set pulse = chan.PulseGenerator
```

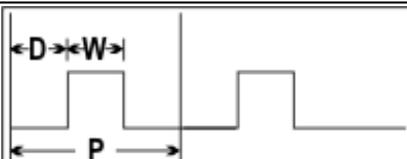
Each pulse generator is specified in the Pulse Generator properties.

External Pulse Generators

Beginning with A.09.50, [External Pulse Generators](#) can be used with the [Integrated Pulse Application](#).

Use chan.[PulseGeneratorNames](#) and chan.[PulseGeneratorID](#) to refer to the external pulse generator when setting properties on this (PulseGenerator) Object.

Pulse Definitions



- D = Delay; the time before each pulse begins
- W = Width; the time the pulse is ON
- P = Period; one complete pulse cycle
- W/P = Duty Cycle; the ratio of pulse ON/OFF

Important: If **D + W** is greater than **P**, then undefined PNA behavior results. There is NO error message or warning.

See Also:

- [IF Path Block Diagram.](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [About PNA-X Pulse Measurements](#)
- [Example Programs](#)

Methods	Description
---------	-------------

None

Properties	Interface	Description
------------	-----------	-------------

[See History](#)

Delay	IPulsedGenerator	Sets the pulse delay.
DelayIncrement	IPulsedGenerator	Sets the pulse delay increment.
Invert	IPulsedGenerator3	Sets whether to invert the polarity of the pulse.
Period	IPulsedGenerator	Sets the pulse-period (1/PRF) for ALL PNA-X internal pulse generators.
State	IPulsedGenerator	Turns the specified pulse generator ON and OFF.
SubPointTrigger	IPulsedGenerator2	Enables / Disables subpoint triggering.
TriggerInPolarity	IPulsedGenerator3	Sets the polarity of trigger to which the internal pulse generators will respond when being externally triggered.
TriggerInType	IPulsedGenerator3	Sets the type of trigger to which the internal pulse generators will respond when being externally triggered.
Width	IPulsedGenerator	Sets the pulse width for the specified pulse generator.

IPulseGenerator History

Interface	Introduced with PNA Rev:
IPulseGenerator	7.2
IPulseGenerator2	8.55.09
IPulseGenerator3	9.10

Last Modified:

8-Oct-2010	Added Invert
9-Dec-2009	Added 9.10 commands
20-Jul-2009	Added SubPointTrig
1-Jan-2007	MX New topic

PulseMeasControl Object

Description

Contains the properties for configuring pulse measurements in the PNA-X.

Some of these settings require Opt H08 / 008.

[Learn about Integrated Pulse Application.](#)

Accessing the PulseMeasControl object

```
Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel
Dim pulse as PulseMeasControl
Set pulse = chan.PulseMeasControl
```

The following shows how to make two new settings introduced with A.09.50:

- **Master Pulse Trigger** - to select an external pulse generator as the master trigger, use `IPathElement.Value` and using "PulseTrigInput" = "Internal" or "External". However, there is no way to tell which external pulse generator is selected.
- To select an external pulse generator for a receiver, use [IPathElement.Value](#) commands. For example, to select an external pulse generator for RCVR A, send "IFGateA","RearPanel".

See Also:

- [IF Path Block Diagram.](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods	Description
---------	-------------

None

Properties	Interface	Description
------------	-----------	-------------

[See History](#)

AutoCWSweepTime	IPulseMeasControl2	Sets the state of automatic CW sweep time in Pulse Profile mode.
AutoDetection	IPulseMeasControl	Automatically or manually set pulse mode (Narrowband or Wideband)
AutoIFBandWidth	IPulseMeasControl	Autoselect the IFBW
AutoIFGain	IPulseMeasControl	For future use.
AutoOptimizePRF	IPulseMeasControl	Auto-optimize pulse clock period
AutoPulseTiming	IPulseMeasControl	Autoselect Width and Delay
AutoSelectPulseGen	IPulseMeasControl	Autoselect Pulse Generators
MasterFrequency	IPulseMeasControl2	Sets the pulse repetition frequency (PRF) for ALL internal pulse generators.
MasterPeriod	IPulseMeasControl2	Sets the period for ALL internal pulse generators.
MasterWidth	IPulseMeasControl2	Sets the pulse width for ALL internal pulse generators.
Parent	IPulseMeasControl	Returns the channel object
PulseMeasMode	IPulseMeasControl	Select Pulse Measurement selection
SoftwareGateState	IPulseMeasControl2	This setting is used for troubleshooting purposes.
WideBandDectionState	IPulseMeasControl	Select Narrowband or Wideband pulse detection.

IPulseMeasControl History

Interface	Introduced with PNA Rev:
IPulseMeasControl	9.2
PulseMeasControl2	9.40

Last Modified:

16-Feb-2012	Fixed name
3-May-2011	Added master commands
11-Mar-2010	MX New topic

RxLevelingConfiguration Object

Description

Contains the properties for configuring Receiver Leveling in the PNA.

Accessing the RxLevelingConfiguration object

```
Dim app as AgilentPNA835x.Application  
Dim chan as Channel  
Set chan = app.ActiveChannel  
Dim RxLevel as RxLevelingConfiguration  
Set RxLevel = chan.RxLevelingConfiguration
```

See Also:

- [About Receiver Leveling](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Program](#)

Methods	Description
---------	-------------

None

Properties	Interface	Description
------------	-----------	-------------

[See History](#)

FastMode	IRxLevelingConfiguration	Select separate IFBW for leveling sweeps
FrequencyType	IRxLevelingConfiguration3	Frequency range to use for receiver leveling.
IterationNumber	IRxLevelingConfiguration	Max number of leveling sweeps
LastLevelingAsSPC	IRxLevelingConfiguration2	Turn ON Source Power Cal using latest correction data
LevelingIFBW	IRxLevelingConfiguration	IFBW for leveling sweeps
PowerMax	IRxLevelingConfiguration	Max power for safe mode

PowerMin	IRxLevelingConfiguration	Min power for safe mode
PowerOffset	IRxLevelingConfiguration	Offset power for external components
PowerStep	IRxLevelingConfiguration	Power step for safe mode
ReferenceReceiver	IRxLevelingConfiguration	Select a receiver
SafeMode	IRxLevelingConfiguration	Enable safe mode
State	IRxLevelingConfiguration	Enable receiver leveling
Tolerance	IRxLevelingConfiguration	Tolerance value for leveling sweeps

ReceiverLevelingConfiguration History

	Interface	Introduced with PNA Rev:
	IRxLevelingConfiguration	8.5
	IRxLevelingConfiguration2	9.30
	IRxLevelingConfiguration3	9.50

Last Modified:

12-Mar-2012	Added IRxLevelingConfiguration3
16-Sep-2010	Added IRxLevelingConfiguration2
13-Feb-2009	MX New topic

SCPIStringParser Object

Description

Provides the ability to send a SCPI command from within the COM command. The two commands differ in the following ways:

Execute - will not return an error unless the Execute command itself fails, which is unlikely.

Otherwise, you are required to read the SCPI error queue for errors that were caused by the SCPI command. The Execute command operates with minimal interference between you, the programmer, and the SCPI parser. It does not presume how you want to handle errors: handle by ignore, handle by reading the status byte, etc. This command was defined because automation engines like VB throw runtime errors when a COM method returns a failed HRESULT.

Parse - parses the input command, and then reads the SCPI error queue until the queue is empty. If the queue contains errors, Parse returns a failed HRESULT (E_NA_BAD_SCPI_EXECUTE). It then creates an IErrorInfo object and bundles the error numbers and descriptions into the error object. This object is available so that you can detect the failed HRESULT and interrogate the errorInfo object for more details.

The SCPIStringParser Methods can NOT be used with:

- [SCPI Status Reporting](#). However, the *OPC? will work.
- Transferring Binary (block) data with commands such as: [MMEM:TRAN](#) or [CALC:DATA](#) with [Format:Data](#) set to Real32 or Real64

Accessing the ScpiStringParser object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim SCPI As IScpiStringParser
Set SCPI = app.ScpiStringParser
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)
- [See an example](#) of how to return error information when using the [Parse method](#).

Methods	Interface	Description
Parse	ISCPIStringParser	Provides the ability to send a SCPI command from within the COM command.

[Execute](#) **ISCPIStringParser2** Does not convert scpi errors. Use :SYST:ERR?

Properties

None

History

Interface	Introduced with PNA Rev:
ISCPIStringParser	1.0
ISCPIStringParser2	3.0

Segment Object

Description

Contains the methods and properties that affect a sweep segment.

Note: All of these properties are shared with at least one of the following objects: Channel, Cal Set, PowerSensorCalFactorSegment, or PowerLossSegment.

Accessing a Segment object and setting Segment Properties

You can get a handle to a sweep segment through the segments collection.

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim segs As ISegments
Set segs = app.ActiveChannel.Segments

segs.Add(1)
segs(1).NumberOfPoints = 30
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Segment Sweep](#)
- [Example Programs](#)

Methods

None

Properties	Interface	Description
centerFrequency	ISegment	Sets or returns the center frequency of the segment. Shared with the Channel Object
DwellTime	ISegment	Dwell time value. Shared with the Channel Object

FrequencySpan	ISegment	Sets or returns the frequency span of the segment. Shared with the Channel Object
IFBandwidth	ISegment	Sets or returns the IF Bandwidth of the segment. Shared with the Channel Object and Cal Set object.
NumberOfPoints	ISegment	Sets or returns the Number of Points of the segment. Shared with the Channel Object
SegmentNumber	ISegment	Returns the number of the current segment.
StartFrequency	ISegment	Sets or returns the start frequency of the segment. Shared with the Channel Object
State	ISegment	Turns On or OFF a segment.
StopFrequency	ISegment	Sets or returns the stop frequency of the segment. Shared with the Channel Object
SweepTime	ISegment2	Sets or returns the sweep time of the segment. Shared with the Channel Object
TestPortPower	ISegment	Sets or returns the RF power level of the segment. Shared with the Channel Object

ISegment History

Interface	Introduced with PNA Rev:
ISegment	1.0
ISegment2	7.1

Last modified:

9/29/06 MQQ Added Sweep time

Segments Collection

Description

The segment collection provides a mechanism for iterating through the sweep segments of a channel. Sweep segments are a potentially faster method of sweeping the analyzer through only the frequencies of interest. Learn more about [Segment Sweep](#).

Accessing the Segments collection

There are two paths to the Segments Collection:

1. From the [Channel](#) object
2. From the [FOMRange](#) object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim segs As ISegments
Set segs = app.ActiveChannel.Segments
```

or

```
Set segs = app.ActiveChannel.FOM.FOMRange(1).Segments
```

See Also:

- [Segment Object](#) to learn how to set the properties for individual segments.
- [Collections in the Analyzer](#)
- [The PNA Object Model](#)

- [Example Programs](#)

Methods	Interface	Description
	See History	
Add	ISegments	Adds an item to either the Segments collection.
Item	ISegments	Use to get a handle to a segment in the collection..
GetAllSegments	ISegments5	Downloads a segment table from the PNA
Remove	ISegments	Removes an item from a collection of objects.
SetAllSegments	ISegments2	Uploads a segment table to the PNA.

Properties		Description
AllowArbitrarySegments	ISegments3	Enables the setup of arbitrary segment sweep
Count	ISegments	Returns the number of items in a collection of objects.
IF Bandwidth Option	ISegments	Enables the IFBandwidth to be set on individual sweep segments.
Parent	ISegments	Returns a handle to the current naNetworkAnalyzer application..
Source Power Option	ISegments	Enables setting the Source Power for a segment.
SweepTimeOption	ISegments4	Enables the Sweep time or Dwell time to be set independently on sweep segments.

ISegments History

Interface	Introduced with PNA Rev:
ISegments	1.0
ISegments2	3.5
ISegments3	4.2
ISegments4	7.1
ISegments5	8.6

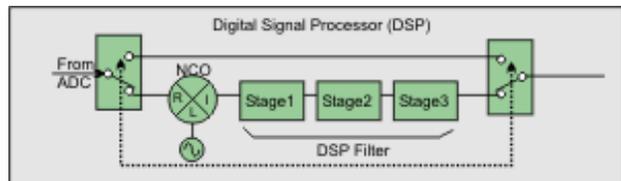
Last modified:

23-Jul-2009	Added more description
28-Apr-2009	Added GetAllSegments
8-Mar-2007	Modified access via fom
9/29/06	Added ISegments4

SignalProcessingModuleFour Object

Description

Contains the properties for configuring the DSP (digital filters) in the PNA-X.



[See the entire IF Path Block diagram.](#)

DSP Version 5 Notes

Programs that control these settings, or state files that are saved, will yield different results when run or recalled on PNAs with DSP 4 versions versus DSP 5 Versions.

Stage 2 settings are IGNORED with DSP 5 Versions.

[Learn more about DSP Versions](#)

Accessing the SignalProcessingModuleFour object

```
Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel
Dim digFilter as SignalProcessingModuleFour
Set digFilter = chan.SignalProcessingModuleFour
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [About PNA-X Pulse Capabilities](#)
- [Example Programs](#)

Methods	Description
None	

Properties	Interface	Description
	See History	
ADCCaptureMode	ISPM4	Sets ADC capture mode: auto or manual
FilterErrors	ISPM4	Returns errors with manual digital filter settings
FilterMode	ISPM4	Sets digital filter mode:auto or manual
Stage1Coefficients	ISPM4	Sets Stage1Coefficients
Stage1Frequency	ISPM4	Sets Stage1 NCO frequency
Stage1MaximumCoefficient	ISPM4	Returns the maximum value of any single stage1 coefficient.
Stage1MaximumCoefficientCount	ISPM4	Returns the maximum number of Stage1 coefficients.
Stage1MaximumCoefficientSum	ISPM4	Returns the maximum sum of all Stage1 coefficients.
Stage1MinimumCoefficientCount	ISPM4	Returns the minimum number of Stage1 coefficients
Stage2Coefficients	ISPM4	Sets Stage2 Coefficients
Stage2MaximumCoefficient	ISPM4	Returns the maximum value of any single stage2 coefficient.
Stage2MaximumCoefficientCount	ISPM4	Returns the maximum number of Stage2 coefficients
Stage2MaximumCoefficientSum	ISPM4	Returns the maximum sum of all Stage2 coefficients.
Stage2MinimumCoefficientCount	ISPM4	Returns the minimum number of Stage2 coefficients
Stage3Coefficients	ISPM42	Sets and returns Stage3Coefficients.
Stage3FilterType	ISPM4	Sets and returns stage3 filter type
Stage3FilterTypes	ISPM4	Returns the names of supported types of Stage3 filters.
Stage3MaximumCoefficient	ISPM42	Returns the maximum number of coefficients for Stage3.
Stage3MaximumCoefficientCount	ISPM42	Returns the maximum number of Stage3coefficients

<u>Stage3MinimumCoefficient</u>	ISPM42	Returns the minimum value of any single stage3coefficient.
<u>Stage3MinimumCoefficientCount</u>	ISPM42	Returns the minimum number of Stage3coefficients
<u>Stage3Parameter</u>	ISPM4	Sets and returns the parameter value of the current filter type.
<u>Stage3ParameterMaximum</u>	ISPM4	Returns maximum parameter value for the current filter type.
<u>Stage3ParameterMinimum</u>	ISPM4	Returns minimum parameter value for the current filter type.
<u>Stage3Parameters</u>	ISPM4	Returns the names of parameters for the current filter type.

ISignalProcessingModuleFour History

Interface	Introduced with PNA Rev:
ISignalProcessingModuleFour2	9.90
ISignalProcessingModuleFour	7.2

Last Modified:

5-Jan-2007 MX New topic

SMCType Object

Description

Contains the methods and properties to perform an Scalar Measurement Calibration for the Frequency Converter Application (option 083).

Accessing the SMCType object

See an example which [creates and calibrates an SMC measurement](#).

You can also do the following:

```
Set app = CreateObject("AgilentPNA835x.Application")

Set CalMgr = app.GetCalManager

Set guidedCal = CalMgr.CreateCustomCalEx(1)

Set SMC = guidedCal.CustomCalConfiguration

SMC.ConnectorType(1) = "APC 3.5 male"
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods	Interface	Description
	See History	
AcquireStep	ISMCType	Acquire the measurement data for the specified step in the calibration process.
GenerateErrorTerms	ISMCType	Generates the error terms for the calibration.
GenerateSteps	ISMCType	Returns the number of steps required to complete the calibration.
GetStepDescription	ISMCType	Returns the description of the specified step calibration process.
ImportDataSet	ISMCType4	Imports separate power meter data for SMC cal.

[Initialize](#) ISMCType Begins a calibration.

Properties

Description

[AutoOrient](#) ISMCType Sets ECAL module automatic orientation ON or OFF.

[CalibrationPort](#) ISMCType Sets or returns the calibration source port for the calibration.

[CalKitType](#) ISMCType Sets and returns a calibration kit type for calibration.

[CompatibleCalKits](#) ISMCType Returns a list of cal kits that are compatible with the connector type for the specified port.

[ConnectorType](#) ISMCType Sets or queries the connector type for the specified port.

[Do2PortEcal](#) ISMCType **Superseded** - Replaced by [CalKitType](#)
Specify ECAL or Mechanical calibration.

[EcalCharacterization](#) ISMCType **Superseded** - Replaced by [CalKitType](#)
Specifies the characterization data within an ECal module to be used for the calibration.

[EcalOrientation](#) ISMCType Specifies which port of the ECal module is connected to which port of the PNA when the AutoOrient property = False.

[EnableLOPowerCal](#) ISMCType5 Enable LO Power Cal

[FixedDelay](#) ISMCType5 Set and return the known delay through the calibration mixer.

[MixerCharacterizationFile](#) ISMCType5 Set the filename of the S2P file used to characterize the calibration mixer.

[NetworkFilename](#) ISMCType2 Specifies the S2P filename to embed or de-embed on the input or output of your mixer measurement.

[NetworkMode](#) ISMCType2 Embed (add) or de-embed (remove) circuit network effects on the input and output of your mixer measurement.

[OmitIsolation](#) ISMCType **Superseded** - Replaced by [SetIsolationPaths](#) and [GetIsolationPaths](#)
Sets and returns whether Isolation portion of the calibration will be performed or not.

[SeparatePowerCal](#) ISMCType4 Use a Thru standard or to use two power sensor connections during the SMC power cal

[ThruCalMethod](#)

ISMCType **Superseded** - Replaced by [PathThruMethod Property](#)

Sets and returns the method for performing the thru portion of the calibration.

[ValidConnectorTypes](#)

ISMCType Returns a list of connector types for which there are calibration kits.

ISMCType History

Interface	Introduced with PNA Rev:
ISMCType	3.5
ISMCType2	6.0
ISMCType4	9.0

SourcePowerCalibrator Object

Description

This object is a child of the Application object and is a vehicle for performing source power calibrations.

Accessing the SourcePowerCalibrator Object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim ispc As ISourcePowerCalibrator
Set ispc = app.SourcePowerCalibrator
```

Note: If you see this error:

The target NA stimulus channel has not been set. With the SourcePowerCalibrator object, you must first call SetCalInfoEx with the channel number. SetCalInfoEx must not be made against a temporary copy of the SourcePowerCalibrator object.

Or this error:

Channel not found.

It is either because you set the wrong channel number in the SetCalInfoEx call or because the SourcePowerCalibrator object is unique in that you **MUST** use the **SAME** instance of the SourcePowerCalibrator object for every property that is set. Do this by naming/setting a variable as in this vbs example:

```
set sourcepowercalibrator = app.sourcepowercalibrator
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Note: Interface **ISourcePowerCalibrator** is abbreviated as **ISPC** in the following table.

(**Bold** Methods or Properties provide access to a child object)

Methods	Interface	Description
	See History	
AbortPowerAcquisition	ISPC	Aborts a source power cal acquisition sweep that is currently in progress.
AcquirePowerReadings	ISPC	Superseded with AcquirePowerReadingsEx
AcquirePowerReadingsEx	ISPC4	Initiates a source power cal acquisition.
ApplyPowerCorrectionValues	ISPC	Superseded with ApplyPowerCorrectionValuesEx
ApplyPowerCorrectionValuesEx	ISPC5	Applies correction values after completing a source power cal acquisition sweep. Optionally perform a calibration of the reference receiver used in the source power cal.
CheckPower	ISPC2	Measures power at a specific frequency. Used to test power level before and/or after applying a source power calibration.
LaunchPowerMeterSettingsDialog	ISPC2	Launches the Power Meter Settings dialog on the PNA.
SetCalInfo	ISPC	Superseded with SetCalInfoEx Method
SetCalInfo2	ISPC3	Superseded with SetCalInfoEx Method
SetCalInfoEx Method	ISPC4	Specifies the channel and source port to be used for the source power calibration.
SetPowerAcquisitionDevice	ISPC3	Sets the power sensor channel (A or B) to be used. This method is ONLY necessary when performing an SMC calibration.
Properties	Interface	Description
CalPower	ISPC	Specifies the power level that is expected at the desired reference plane.
IterationsTolerance	ISPC3	Sets the maximum desired deviation from the sum of the test port power and the offset value.
LastCalPassedTolerance	ISPC7	Returns pass / fail status of the user-specified tolerance limits on the target cal power.
MaximumIterationsPerPoint	ISPC3	Specifies maximum number of readings to take at each data point for iterating the source power.

<u>PowerAcquisitionDevice</u>	ISPC2	Specifies the power sensor channel (A or B) that is currently selected for use at a specific frequency.
<u>PowerLossSegments (collection)</u>	ISPC2	Collection for iterating through the segments of the power loss table used in source power calibration.
<u>PowerMeterGPIBAddress</u>	ISPC	Specifies the GPIB address of the power meter.
<u>PowerMeterInterfaces</u>	ISPC6	Collection for getting a handle to the available power meters.
<u>PowerSensors (collection)</u>	ISPC2	Collection for iterating through the PowerSensor objects which are connected to the power meter for a source power cal.
<u>ReadingsPerPoint</u>	ISPC	Specifies the maximum power readings for power meter settling.
<u>ReadingsTolerance</u>	ISPC3	Power meter settling tolerance value.
<u>USBPowerMeterCatalog</u>	ISPC6	Returns a list of USB power meters that are connected to the PNA.
<u>UsePowerLossSegments</u>	ISPC	Specifies if subsequent calls to the AcquirePowerReadings method will make use of the loss table (PowerLossSegments).
<u>UsePowerSensorFrequencyLimits</u>	ISPC	Specifies if subsequent calls to the AcquirePowerReadings method will make use of power sensor frequency checking capability.

ISourcePowerCalibrator History

Interface	Introduced with PNA Rev:
ISourcePowerCalibrator	2.0
ISourcePowerCalibrator2	3.5
ISourcePowerCalibrator3	4.0
ISourcePowerCalibrator4	6.2
ISourcePowerCalibrator5	7.2
ISourcePowerCalibrator6	7.5
ISourcePowerCalibrator7	9.2

SweptIMD Object

Description

Controls the Swept IMD Application settings.

See [Properties to set for each sweep type](#).

Accessing the SweptIMD object

```
Dim app as AgilentPNA835x.Application

Set app = CreateObject("AgilentPNA835x.Application")

app.CreateCustomMeasurementEx(1, "SweptIMD", "PwrMain", 1)

Dim IMD

Set IMD = app.ActiveChannel.CustomChannelConfiguration
```

See Also:

- **Example Program** [Create and Cal a Swept IMD Measurement](#)
- Use std channel commands to set [source](#) and [receiver attenuation](#).
- [SweptIMDCal Object](#)
- [About the Swept IMD Application](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)

Method	Interface	Description
	See History	
SetPortMap	ISweptIMD	Sets port mapping
Property	Interface	Description
CompositeNormalizationMode	ISweptIMD	Sets and returns the method by which CTB and CSO calculations are performed.
CompositeNormalizedCSOPower	ISweptIMD	Sets and returns the CSO Power.
CompositeNormalizedCTBPower	ISweptIMD	Sets and returns the CTB Power.

CoupleTonePower	ISweptIMD ON OFF state of power coupling for F1 and F2.
CSONumDistortionProducts	ISweptIMD Sets the “N = number of distortion products” value for the calculation of the CSO parameter.
CSOOffset Property	ISweptIMD Sets and returns the offset that is applied to CSO measurements.
CTBOffset Property	ISweptIMD Sets and returns the offset that is applied to CTB measurements.
CTBXMomodNumCarriers	ISweptIMD Sets the “N = Total number of carriers” value used in the calculation of the XMOD and CTB parameter.
DeltaFrequency	ISweptIMD Fixed tone spacing value.
DeltaFrequencyStart	ISweptIMD Start spacing of the main tones.
DeltaFrequencyStop	ISweptIMD Stop spacing of the main tones.
DeviceInputPort	ISweptIMD Reads input port map
DeviceOutputPort	ISweptIMD Reads output port map
EqualTonePower	ISweptIMD2 Superseded with LevelingMethod Set Equal Tone Power state.
F1Frequency	ISweptIMD Frequency of the F1 tone.
F2Frequency	ISweptIMD Frequency of the F1 tone.
FrequencyCenter	ISweptIMD Center frequency of the main tones.
FrequencyCenterCenter	ISweptIMD Sweep center frequency when sweeping the main tones.
FrequencyCenterSpan	ISweptIMD Frequency span when sweeping the main tones.
FrequencyCenterStart	ISweptIMD Start frequency when sweeping the main tones.
FrequencyCenterStop	ISweptIMD Stop frequency when sweeping the main tones.
HighestOrderProduct	ISweptIMD Reads the highest product that can be measured by SweptIMD.
IMToneIFBandwidth	ISweptIMD IF Bandwidth for measurement of the intermodulation products.
LevelingMethod	ISweptIMD Set power leveling mode
MainToneIFBandwidth	ISweptIMD IF Bandwidth for measurement of the Main tones.
SweepType	ISweptIMD Type of sweep for a Swept IMD measurement.

TonePower	ISweptIMD	Power level of the Main Tones.
TonePowerSetAt	ISweptIMD2	Superseded with LevelingMethod Set power level at DUT Input or Output
TonePowerStart	ISweptIMD	Start power level of the Main tones.
TonePowerStop	ISweptIMD	Stop power level of the Main tones.

See Also

- [RoleDevice Property](#) to use an external source for f2.

The following commands are relevant for each [IMD sweep type](#):

0 - naIMDToneCWSweep

- [F1Frequency Property](#)
- [F2Frequency Property](#)
- [FrequencyCenter Property](#)
- [DeltaFrequency Property](#)
- [TonePower \(1,2\) Property](#)

1 - naIMDTonePowerSweep

- [F1Frequency Property](#)
- [F2Frequency Property](#)
- [FrequencyCenter Property](#)
- [DeltaFrequency Property](#)
- [TonePowerStart Property](#)
- [TonePowerStop Property](#)

2 - naIMDToneCenterFreqSweep

- [FrequencyCenterStart Property](#)
- [FrequencyCenterStop Property](#)
- [FrequencyCenterCenter Property](#)

- [FrequencyCenterSpan Property](#)
- [DeltaFrequency Property](#)
- [TonePower \(1,2\) Property](#)

3 - naIMDDeltaFrequencySweep

- [DeltaFrequencyStart Property](#)
- [DeltaFrequencyStop Property](#)
- [FrequencyCenter Property](#)
- [TonePower \(1,2\) Property](#)

4 - naIMDToneSegmentSweep (Not available for IMDx)

- [DeltaFrequency Property](#)
- [TonePower \(1,2\) Property](#)
- Also use standard [segment sweep commands](#)

5 - naLOPowerSweep (IMDx ONLY)

- converter.[LOStartPower Property](#)
- converter.[LOStopPower Property](#)

ISweptIMD History

Interface	Introduced with PNA Rev:
ISweptIMD	8.33
ISweptIMD2	9.40
ISweptIMD3	9.80

Last Modified:

- 6-Nov-2012 Added Leveling method
- 18-Jun-2012 Added links to source and receiver attenuation
- 18-Apr-2012 Added link to RoleDevice
- 5-May-2011 Added tone power settings
- 11-Aug-2009 Added limited port map
- 25-Mar-2009 Fixed Access example
- 19-Aug-2008 MX New topic

SweptIMDCal Object

Description

Sets properties that are unique to a Swept IMD Cal (opt 087).

Use the [Guided Calibration commands](#) for the remaining commands to perform a Swept IMD Cal

Accessing the SweptIMDCal object

```
Dim app as AgilentPNA835x.Application

Set IMDCal = pna.GetCalmanager.CreateCustomCalEx\(channelNum\)

Set IMDCalExtension = IMDCal.CustomCalConfiguration

IMDCalExtension.PowerLevel = 5
```

See Also:

- [Example Program Create and Cal a Swept IMD Measurement](#)
- [SweptIMD Object](#)
- [About Swept IMD measurements](#)
- [The PNA Object Model](#)
- [PNA Automation Interfaces](#)

Method	Interface	Description
	See History	

None

Property	Interface	Description
CalibrationFrequencies	ISweptIMDCal	Perform the source power cal at the center frequencies or at all main tone frequencies.
CalMethod	ISweptIMDCal	Sets the method by which the match-correction portion of an IMD cal is performed
EnableLOPowerCal	ISweptIMDCal2	Enable or disable an LO power cal with an IMDx calibration.
Include2ndOrderProduct	ISweptIMDCal	Include the second order products in the calibration.

MaxProduct	ISweptIMDCal	Sets and returns the maximum intermod product frequencies to be calibrated.
PowerLevel	ISweptIMDCal	Set and read the power level of the source power cal.
PowerSensorCalKitType	ISweptIMDCal	Set and read the cal kit to be used for port 1 adapter compensation.
PowerSensorConnectorType	ISweptIMDCal	Superseded Use GuidedCal. PowerSensorConnectorType

ISweptIMD History

Interface	Introduced with PNA Rev:
ISweptIMDCal	8.33
ISweptIMDCal2	8.55

Last Modified:

9-Sep-2008 MX New topic

TestsetControl Object

Description

A TestsetControl object is used to control one of the [supported test sets](#). Only one external test set can be controlled by the PNA at any time. The Testset Control object appears as an item in the ExternalTestsets collection, which in turn is a property of the main application object.

If the specified test set is not connected to the PNA or is not ON, then setting [Enabled](#) = True will return an error. All other properties can be set even if the test set is not connected.

Note: The ONLY way to load a test set configuration file is by sending the [testsets.Add](#) method. There is no method to query the test set type. See an [example program](#).

Accessing a TestsetControl object

The [ExternalTestsets collection](#) is a property of the main **Application** Object. You can obtain a handle to a testset object by specifying an item in the collection.

Visual Basic Example

```
Dim pna
Dim testsets As ExternalTestsets
Dim tset1 As TestsetControl
Set pna = CreateObject("AgilentPNA835x.Application")
Set testsets = pna.ExternalTestsets
Set tset1 = testsets(1)
' make COM calls on tset1 object
End Sub
```

See Also:

- [E5091A Testset Object](#)
- [About External Testset Control](#)
- [ExternalTestset Control Example](#)
- [ExternalTestsets Collection](#)
- [The PNA Object Model](#)

Methods	Interface	Description
---------	-----------	-------------

([See history](#))

None

Properties		Description
------------	--	-------------

ControlLines	IExternalTestset	Sets the control lines of the specified Test set.
Enabled	IExternalTestset	Enables and disables (ON/OFF) the port mapping and control line output of the specified test set.
ID	IExternalTestset	Returns the test set ID number.
Label	IExternalTestset	Returns the label on a given channel for the specified test set.
NumberOfPorts	IExternalTestset	Reads the number of ports that are on the specified test set.
OutputPorts	IExternalTestset	Sets or returns the port mappings for ALL ports.
PortCatalog	IExternalTestset	Returns the selections available for a given logical port.
SelectPort	IExternalTestset	Sets and returns the logical port value.
ShowProperties	IExternalTestset	Turns status bar display of test set properties on or off.
Type	IExternalTestset	Returns the test set model.

ExternalTestset History

	Interface	Introduced with PNA Rev:
--	-----------	--------------------------

	IExternalTestset	6.0
	IExternalTestset	6.0

Trace Object

Description

The Trace object controls how the measurement data is displayed. You can control scale, reference position, and value from the Trace Object.

Accessing a Trace object

There are several ways to get a handle to a trace.

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim trace As Trace
```

Then you can do any of the following:

```
Set trace = app.NAWindows(1).traces(1)
```

```
set trace = app.NAWindows.item(1).ActiveTrace
```

```
set trace = app.ActiveNAWindow.traces.item(1)
```

```
set trace = app.ActiveNAWindow.ActiveTrace
```

```
Set trace = app.Measurements(1).trace
```

```
Set trace = app.ActiveMeasurement.trace
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Traces, Channels, and Windows on the PNA](#)
- [Example Programs](#)

Methods	Interface	Description
---------	-----------	-------------

Autoscale	ITrace	Autoscales the trace or all of the traces in the selected window. Shared with the NAWindow Object
Move	ITrace3	Moves a trace from one window to another.

Property		Description
----------	--	-------------

Measurement	ITrace2	Returns a measurement handle from trace object.
Name	ITrace	Sets or returns the trace name
ReferencePosition	ITrace	Sets or returns the Reference Position of the active trace.
ReferenceValue	ITrace	Sets or returns the value of the Y-axis Reference Level of the active trace.
YScale	ITrace	Sets or returns the Y-axis Per-Division value of the active trace.

ITrace History

Interface	Introduced with PNA Rev:
ITrace	1.0
ITrace2	9.40
ITrace3	9.50

Traces Collection

Description

Child of the **Application** Object. A collection that provides a mechanism for getting a handle to a trace or iterating through the traces in a window.

Accessing the Traces collection

Get a handle to the traces collection through the NaWindows collection. The following example sets the variable **trcs** to the collection of traces in window 1 of the NaWindows collection.

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim trcs As traces
Set trcs = app.NAWindows(1).traces
```

See Also:

- [Trace Object](#)
- [Collections in the Analyzer](#)
- [The PNA Object Model](#)
- [Example Programs](#)

	Description
Methods	
Item	Use to get a handle to a trace
Properties	
Count	Returns the number of traces in the collection.
Parent	Returns a handle to the current Application.

Transform Object

Description

Contains the methods and properties that control Time Domain transforms.

Accessing the Transform Object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim trans As Transform
Set trans = app.ActiveMeasurement.Transform
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Time Domain Topics](#)
- [Example Programs](#)

Note: [Sweep Type](#) must be set to Linear before setting Time Domain Transform (state) ON.

Methods	Interface	Description
SetFrequencyLowPass	ITransform	Sets low frequencies for low pass.
Properties		Description
Center	ITransform	Sets or returns the Center time. Shared with the Gating Object
CoupledParameters	ITransform2	Select Transform parameters to couple
DistanceMarkerMode	ITransform2	Sets the measurement type in order to determine the correct marker distance.
DistanceMarkerUnit	ITransform2	Sets the unit of measure for the display of marker distance values.
ImpulseWidth	ITransform	Sets or returns the Impulse Width of Time Domain transform windows.

<u>KaiserBeta</u>	ITransform	Sets or returns the Kaiser Beta of Time Domain transform windows.
<u>Mode</u>	ITransform	Sets the type of transform.
<u>Span</u>	ITransform	Sets or returns the Span time. Shared with the Gating Object
<u>Start</u>	ITransform	Sets or returns the Start time. Shared with the Gating Object
<u>State</u>	ITransform	Turns an Object ON and OFF.
<u>StepRiseTime</u>	ITransform	Sets or returns the Rise time of the stimulus in Low Pass Step Mode.
<u>Stop</u>	ITransform	Sets or returns the Stop time. Shared with the Gating Object

ITransform History

Interface	Introduced with PNA Rev:
ITransform	1.0
ITransform2	4.2

TriggerSetup Object

Description

These properties setup Global triggering that effects the entire PNA application.

Accessing the TriggerSetup object

```
Dim app as AgilentPNA835x.Application
Dim trigSetup as ITriggerSetup
Set trigSetup = app.TriggerSetup
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Triggering in the PNA](#)
- [Example Programs](#)

Methods	Interface	Description
	See History (below)	

None

Properties		Description
AcceptTriggerBeforeArmed	ITriggerSetup2	Allows a trigger signal to be remembered and then used when the PNA becomes armed (ready to be triggered).
ExternalTriggerConnectionBehavior	ITriggerSetup	Configures the external triggering signal for the PNA
ReadyForTriggerPolarity	ITriggerSetup3	Specifies the polarity of Ready for Trigger output.
Scope	ITriggerSetup	Determines whether a trigger signal affects a single channel or all channels in the PNA.

[Source](#)

ITriggerSetup Sets or returns the source of triggering in the PNA.

[TriggerOutputEnabled](#)

ITriggerSetup2 **Superseded**

Enables the PNA to send trigger signals out the rear-panel TRIGGER OUT connector.

ITriggerSetup History

Interface	Introduced with PNA Rev:
ITriggerSetup	4.0
ITriggerSetup2	4.2
ITriggerSetup3	7.50.2 and 8.2

VMC Type Object

Description

Contains the methods and properties to perform a Vector Measurement Calibration for the Frequency Converter Application (option 083).

Accessing the VMCType object

See an example which [creates and calibrates a VMC measurement](#).

You can also do the following:

```
Set app = CreateObject("AgilentPNA835x.Application")

Set CalMgr = app.GetCalManager

Set guidedCal = CalMgr.CreateCustomCalEx(1)

Set VMC = guidedCal.CustomCalConfiguration

VMC.ConnectorType(1) = "APC 3.5 male"
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)

Methods	Interface	Description
	See History	
AcquireStep	IVMCType	Acquire the measurement data for the specified step in the calibration process.
GenerateErrorTerms	IVMCType	Generates the error terms for the calibration.
GenerateSteps	IVMCType	Returns the number of steps required to complete the calibration.
GetStepDescription	IVMCType	Returns the description of the specified step in the calibration process.
Initialize	IVMCType	Begins a calibration.

Properties		Description
AutoOrient	IVMCType	Sets ECAL module automatic orientation ON or OFF.
CalKitType	IVMCType	Sets and returns a calibration kit type for calibration.
CharacterizeMixerOnly	IVMCType	Sets and returns whether to perform a mixer characterization ONLY or full 2-port calibration.
CharFileName	IVMCType	Specifies the .S2P mixer characterization file name.
CharMixerReverse	IVMCType2	Specifies the direction in which to characterize the calibration mixer.
CompatibleCalKits	IVMCType	Returns a list of cal kits that are compatible with the connector type for the specified port.
ConnectorType	IVMCType	Sets or queries the connector type for the specified port.
Do1PortEcal	IVMCType	Superseded with CalKitType Specify ECAL or Mechanical calibration for the mixer characterization portion of a VMC calibration.
Do2PortEcal	IVMCType	Superseded with CalKitType Specify ECAL or Mechanical calibration for the 2-port calibration portion of a VMC calibration.
EcalCharacterization	IVMCType	Superseded with CalKitType Specifies the characterization data within an ECal module to be used for the calibration.
EcalOrientation1Port	IVMCType	For Mixer Characterization ONLY - Specifies which port of the ECal module is connected to which port of the PNA
EcalOrientation2Port	IVMCType	For full 2-port VMC cal - Specifies which port of the ECal module is connected to which port of the PNA
EnableLOPowerCal	IVMCType4	Perform LO Power Cal
LoadCharFromFile	IVMCType	Specifies and loads a mixer characterization (S2P) file.
NetworkFilename	IVMCType3	Specifies the S2P filename to embed or de-embed on the input or output of your mixer measurement.
NetworkMode	IVMCType3	Embed (add) or de-embed (remove) circuit network effects on the input and output of your mixer measurement.
OmitIsolation	IVMCType	Superseded - Replaced by SetIsolationPaths and

[GetIsolationPaths](#)

Sets and returns whether Isolation portion of the calibration will be performed or not.

[ThruCalMethod](#)

IVMCType

Superseded - Replaced by [PathThruMethod Property](#)

Sets and returns the method for performing the thru portion of the calibration.

[ValidConnectorTypes](#)

IVMCType

Returns a list of connector types for which there are calibration kits.

IVMCType History

Interface	Introduced with PNA Rev:
IVMCType	3.5
IVMCType2	3.53
IVMCType3	6.0
IVMCType4	9.1

AcceptTriggerBeforeArmed Property

Description Determines what happens to an EDGE trigger signal if it occurs before the PNA is ready to be triggered. (LEVEL trigger signals are always ignored.) For more information, see [External triggering](#).

VB Syntax *trigsetup.AcceptTriggerBeforeArmed = boolean*

Variable [\(Type\)](#) - Description

trigsetup A [TriggerSetup2](#) (object)

boolean Choose from:

False - A trigger signal is ignored if it occurs before the PNA is ready to be triggered.

True - A trigger signal is remembered and then used when the PNA becomes armed (ready to be triggered). The PNA remembers only one trigger signal.

Return Type Boolean

Default False

Examples `trigsetup.AcceptTriggerBeforeArmed = True 'Write`

`atba = trigsetup.AcceptTriggerBeforeArmed 'Read`

C++ Syntax HRESULT get_AcceptTriggerBeforeArmed(BOOL *pVal);

HRESULT put_AcceptTriggerBeforeArmed(BOOL newVal);

Interface ITriggerSetup2

AcquisitionDirection Property

Description	Specifies the direction of each part of a 2-port calibration.
VB Syntax	<code>cal.AcquisitionDirection = value</code>
Variable	(Type) - Description
<i>cal</i>	A Calibrator (object)
<i>value</i>	(enum NADirection) - Choose from: 0 - naForward - measures the forward direction 1 - naReverse - measures the reverse direction
Return Type	Long Integer
Default	naForward
Examples	<code>cal.AcquisitionDirection = naForward</code>
C++ Syntax	HRESULT AcquisitionDirection(tagNADirection dir);
Interface	ICalibrator

AcquisitionMode Property

Description Set and read the method by which gain compression data is acquired.

VB Syntax `gca.AcquisitionMode = value`

Variable [\(Type\)](#) - Description

gca A [GainCompression](#) (object)

value (**NAGCAAcquisitionMode**) Choose from:

- **naSmartSweep (0)** Iterate quickly to find compression point
- **naSweepPowerAtEachFreq2D (1)** Sweep power at each frequency
- **naSweepFreqAtEachPower2D (2)** Sweep frequency at each power level

Return Type Enum

Default **naSmartSweep**

Examples

```
gca.AcquisitionMode = naSmartSweep 'Write'
acqMode = gca.AcquisitionMode 'Read'
```

C++ Syntax HRESULT get_AcquisitionMode(tagNAGCAAcquisitionMode* mode)
HRESULT put_AcquisitionMode(tagNAGCAAcquisitionMode mode)

Interface IGainCompression

Last Modified:

11-Sep-2007 MX New topic

ActiveCalKit Property

Description Returns a handle to the Active CalKit object. The active cal kit is the kit selected for use in **Unguided** calibrations.

You can either **(1)** use the handle directly to access CalKit properties and methods, or **(2)** set a variable to the CalKit object. The variable retains a handle to the original object if another CalKit becomes active.

VB Syntax 1) `app.ActiveCalKit.<setting>`
or
2) Set `cKit = app.ActiveCalKit`

Variable [\(Type\) - Description](#)

app An [Application](#) (object)

<setting> A CalKit property (or method) and arguments

cKit (object) - A CalKit object

Return Type CalKit object

Default None

Examples

```
Public cKit as calKit
Set cKit = app.ActiveCalKit 'read
```

C++ Syntax HRESULT get_ActiveCalKit (ICalKit * kit)

Interface IApplication

Last Modified:

3-Jun-2008 Added Unguided

ActiveChannel Property

Description	Returns a handle to the Active Channel object. You can either (1) use the handle directly to access channel properties and methods, or (2) set a variable to the channel object. The variable retains a handle to the original channel if another channel becomes active.
VB Syntax	(1) <code>app.ActiveChannel.<setting></code> or (2) Set <code>chan = app.ActiveChannel</code>
Variable	(Type) - Description
<i>chan</i>	A Channel (object)
<i>app</i>	An Application (object)
<i><setting></i>	A channel property (or method) and arguments
Return Type	Channel object
Default	Not applicable
Examples	<pre>1) app.ActiveChannel.Averaging = 1 2) Public chan as Channel Set chan = app.ActiveChannel</pre>
C++ Syntax	HRESULT get_ActiveChannel(IChannel* *pVal)
Interface	IApplication

Active (ExtDev) Property

Description Set and return the state of activation of the device. When [extDev.IOEnable](#) = True, and this command is set to True, the PNA will attempt communication with the external device. An error is returned if communication cannot be verified.

Note: Send this command AFTER sending other External Device settings to avoid communicating with the device before it has been fully configured.

See Also [ExternalDeviceDeActivatePolicy Property](#) - Determines whether External Devices remain activated or are de-activated when the PNA is Preset or when a Instrument State is recalled.

VB Syntax *extDevices.Active* = *value*

Variable [\(Type\)](#) - [Description](#)

extDevices An [ExternalDevice](#) (object)

value (Boolean) Choose from:

True - Device is active.

False - Device is NOT active.

Return Type Boolean

Default False - When configured using the front panel user interface, the device is ON (activated) by default.

Examples

```
extDevices.Active = True 'Write
bool = extDevices.Active 'Read

See example program to configure PMAR device
See example program to configure External Source
```

C++ Syntax HRESULT get_Active(VARIANT_BOOL* value);

HRESULT put_Active(VARIANT_BOOL newVal);

Interface IExternalDevices

17-Sep-2012 Edited description

31-Jul-2009 MX New topic

ActiveMarker Property

Description	Returns a handle to the Active Marker object. You can either (1) use the handle directly to access Marker properties and methods, or (2) set a variable to the Marker object. The variable retains a handle to the original object if another Marker becomes active.
VB Syntax	1) <code>meas.ActiveMarker.<setting></code> or 2) Set <code>mark = meas.ActiveMarker</code>
Variable	(Type) - Description
<code>meas</code>	(object) - An Measurement object
<code><setting></code>	A marker property (or method) and arguments
<code>mark</code>	(object) - A marker object
Return Type	marker object
Default	None
Examples	<pre>Public mark as marker Set mark = meas.ActiveMarker</pre>
C++ Syntax	HRESULT get_ActiveMarker(IMarker** marker)
Interface	IMeasurement

ActiveMeasurement Property

Description	Returns a handle to the Active Measurement object. You can either (1) use the handle directly to access measurement properties and methods, or (2) set a variable to the measurement object. The variable retains a handle to the original measurement.
VB Syntax	1) <code>app.ActiveMeasurement.<setting></code> or 2) Set <code>meas = app.ActiveMeasurement</code>
Variable	(Type) - Description
<code>meas</code>	A Measurement (object)
<code>app</code>	An Application (object)
<code><setting></code>	A measurement property (or method) and arguments
Return Type	Measurement object
Default	None
Examples	<pre>1) app.ActiveMeasurement.Averaging = 1 2) Public meas as Measurement Set meas = app.ActiveMeasurement</pre>
C++ Syntax	<code>HRESULT get_ActiveMeasurement(IMeasurement **ppMeas)</code>
Interface	IApplication

ActiveNAWindow Property

Description	Returns a handle to the Active Window object. You can either (1) use the handle directly to access window properties and methods, or (2) set a variable to the window object. The variable retains a handle to the original window if another window becomes active.
VB Syntax	1) <code>app.ActiveNAWindow.<setting></code> or 2) Set <code>win = app.ActiveNAWindow</code>
Variable	(Type) - Description
<code>win</code>	A NAWindow (object)
<code>app</code>	An Application (object)
<code><setting></code>	A NAWindow property (or method) and arguments
Return Type	A NAWindow object
Default	Not applicable
Examples	<pre>Public win as NAWindow Set win = app.ActiveWindow</pre>
C++ Syntax	HRESULT get_ActiveNAWindow(INAWindow **ppWindow)
Interface	IApplication

ActiveTrace Property

Description	Returns a handle to the Active Trace object. You can either (1) use the handle directly to access trace properties and methods, or (2) set a variable to the trace object. The variable retains a handle to the original trace if another trace becomes active.
VB Syntax	1) <code>win.ActiveTrace.<setting></code> or 2) <code>Set trce = win.ActiveTrace</code>
Variable	(Type) - Description
<i>trce</i>	A Trace (object)
<i>win</i>	An NAWindow (object)
<i><setting></i>	A trace property (or method) and arguments
Return Type	An NAWindow object
Default	None
Examples	<pre>1) win.ActiveTrace.Autoscale 2) Public trce as Trace Set trce = Application.ActiveNAWindow.ActiveTrace</pre>
C++ Syntax	HRESULT get_ActiveTrace(ITrace**pVal)
Interface	INAWindow

ActiveBackground Property

Description Set and return the background color of the active window for the PNA display or hardcopy print.

VB Syntax `colors.ActiveBackground = value`

Variable [\(Type\)](#) - Description

colors A [ComColors](#) (object)

value **(Long Integer)** - RGB color of the ActiveBackground pen.

Convert the three RGB colors to an integer as follows:

```
RGB = R+(G*2^8)+(B*2^16)
```

To find the three RGB values from the [Display Colors dialog](#), click **Change Color**, then **Define Custom Color**.

Return Type Long

Default Display = 0,0,24

Examples

```
R = 10
G = 10
B = 10
RGB = R+(G*2^8)+(B*2^16)
colors.ActiveBackground = RGB 'Write
color = colors.ActiveBackground 'Read
```

C++ Syntax HRESULT get_ActiveBackground(long* pVal);

HRESULT put_ActiveBackground(long newVal);

Interface IComColors2

Last Modified:

22-Feb-2010 MX New topic

ActiveLabels Property

Description Set and return the labels and grid frame colors in the active window for the PNA display or hardcopy print. (Active labels, Grid frame)

VB Syntax `colors.ActiveLabels = value`

Variable [\(Type\)](#) - Description

colors A [ComColors](#) (object)

value **(Long Integer)** - RGB color of the ActiveLabels pen.

Convert the three RGB colors to an integer as follows:

```
RGB = R+(G*2^8)+(B*2^16)
```

To find the three RGB values from the [Display Colors dialog](#), click **Change Color**, then **Define Custom Color**.

Return Type Long

Default Display = 175,175,175

Print = 0,0,0 (Black)

Examples

```
R = 10
G = 10
B = 10
RGB = R+(G*2^8)+(B*2^16)
colors.ActiveLabels = RGB 'Write
color = colors.ActiveLabels 'Read
```

C++ Syntax HRESULT get_ActiveLabels(long* pVal);

HRESULT put_ActiveLabels(long newVal);

Interface IComColors

Last Modified:

7-Aug-2009 MX New topic

ActiveXAxisRange Property

Description For [FCA](#) and [GCX](#) measurements, sets the swept parameter to display on the X-axis. This command does not change the default setting for new traces.

Use [Converter.ActiveXAxisRange](#) to change all existing traces and make the setting the default setting for new traces.

This command is NOT used for NFX, IMDX, and IMSX measurements. Use [Converter.ActiveXAxisRange](#).

VB Syntax `mixer.ActiveXAxisRange = value`

Variable [\(Type\)](#) - Description

mixer A Mixer (**object**)

value (**Enum as MixerStimulusRange**) - Parameter to display on the X-axis.

Choose from:

0 - mixINPUT - Input frequency span

1 - mixLO_1 - First LO frequency span

2 - mixLO_2 - Second LO frequency span

3 - mixOUTPUT - Output frequency span

Return Type Enum

Default OUTPUT

Examples `mixer.ActiveXAxisRange = 1 'Write`

`variable = mixer.ActiveXAxisRange 'Read`

C++ Syntax `HRESULT get_ActiveXAxisRange(tagMixerStimulusRange *Val)`

`HRESULT put_ActiveXAxisRange(tagMixerStimulusRange newVal)`

Interface IMixer3

Last Modified:

27-Sep-2010 Clarification

26-Jan-2009 Corrected enums

ActiveXAxisRange Property

Description Sets the swept frequency range to display on the X-axis for all existing traces and sets the default for all future traces.

VB Syntax *obj.ActiveXAxisRange = value*

Variable [\(Type\)](#) - Description

obj A [Converter Object](#) (for IMDX and NFX measurements)

or

A [Measurement Object](#) (for GCX, SMC, and VMC measurements)

value **(Enum as ConverterStimulusRange)**

Swept stimulus range to display on the X-axis. Choose from:

0 - naInputRange - Input frequency range

1 - naLO1Range - LO 1 frequency range

2 - naLO2Range - LO 2 frequency range

3 - naOutputRange - Output frequency range

4 - naPerMeasurementRange - reserved for future use.

If the specified frequency range is not swept, the default swept range is used.

Return Type Enum

Default Search is performed in the following order until a swept range is found:

1. OUTPUT
2. INPUT (If the OUTPUT is fixed)
3. Number of Points (If ALL ranges are fixed)

Examples

```
conv.ActiveXAxisRange = naInputRange 'Write
```

```
variable = meas.ActiveXAxisRange 'Read
```

C++ Syntax HRESULT get_ActiveXAxisRange(tagConverterStimulusRange range *Val)

HRESULT put_ActiveXAxisRange(tagConverterStimulusRange range
newVal)

Interface IConverter

IMeasurement4

Last Modified:

23-Dec-2011	Added IMeas
9-Sep-2010	Added LO2 range and Modified for all converter apps.
2-Feb-2009	Added converter
26-Jan-2009	Corrected enums

ADCCaptureMode Property

Description Sets and returns the ADC capture mode modeled as a 2-pole switch in the diagram on the [SignalProcessingModuleFour](#) page. The switch either bypasses or routes the IF through the 3-stage digital filter.

VB Syntax `spm4.ADCCaptureMode = value`

Variable [\(Type\)](#) - Description

`spm4` A [SignalProcessingModuleFour](#) (object)

`value` (Enum as NAStates) Capture mode.

naOFF (0) - The digital filters are used to process IF information. The filters can be configured automatically or manually using [FilterMode Property](#).

naON (1) - The digital filters are bypassed and the raw ADC readings are taken directly. With DSP 4 versions, a maximum of 4096 data points per sweep can be acquired.

With DSP 5 versions, the [PNA maximum data points](#) per sweep can be acquired.

[Learn more about DSP Versions.](#)

Return Type Enum

Default OFF

Examples

```
spm4.ADCCaptureMode = 0 'Write
mode = spm4.ADCCaptureMode 'Read
```

C++ Syntax HRESULT get_ADCCaptureMode(tagNAStates* pCaptureMode);
 HRESULT put_ADCCaptureMode(tagNAStates pCaptureMode);

Interface ISignalProcessingModuleFour

Last Modified:

26-Aug-2010 Updated for DSP 5 (A.09.30)

18-Jun-2007 MX New topic

ALCLevelingMode Property

Description Sets and returns the ALC mode for the specified channel and port. Use [GetSupportedALCModes](#) to return a list of valid ALC modes for the PNA.

[Learn more about ALC mode.](#)

VB Syntax *chan.ALCLevelingMode (sourcePort) = value*

Variable [\(Type\)](#) - Description

chan **(object)** - A [Channel](#) object

sourcePort **(long integer)** - The source port for which to make this setting. If ports are [remapped](#), specify the logical port number.

Note: If the port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-x model, then you must use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#).

value **(enum as naALCLevelingMode)** - Choose from:

- **0** naALCInternal
- **1** naALCExternal (E835x Only)
- **2** naALCOpenLoop (PNA-X only)

Return Type Enum

Default naALCInternal

Examples

```
chan.ALCLevelingMode(1) = 'Write
state = chan.ALCLevelingMode(4) 'Read
```

C++ Syntax HRESULT get_ALCLevelingMode(long port, tagNAALCLevelingMode* pVal);

HRESULT put_ALCLevelingMode(long port, tagNAALCLevelingMode newVal);

Interface IChannel9

Last modified:

3-Mar-2009	Removed 3
24-Apr-2008	Added note for string names
30-Apr-2007	Edited for src strings
10/18/06	MX New topic

AllowArbitrarySegments Property

Description Enables you to setup a segment sweep with arbitrary frequencies. The start and stop frequencies of each segment can overlap other segments. Also, each segment can have a start frequency that is greater than its stop frequency which causes a reverse sweep over that segment. Learn more about [Arbitrary Segment Sweep](#).

VB Syntax *segs.AllowArbitrarySegments = value*

Variable [\(Type\)](#) - Description

segs A [Segments](#) collection (**object**)

value (**boolean**)

True - Allows the setup of arbitrary segment sweep.

False - Prevents the setup of arbitrary segment sweep.

Return Type Boolean

Default False

Examples

```
segs.AllowArbitrarySegments = True 'Write
```

```
AllowArbSegs = AllowArbitrarySegments 'Read
```

C++ Syntax HRESULT get_AllowArbitrarySegments(VARIANT_BOOL *pVal)

HRESULT put_AllowArbitrarySegments(VARIANT_BOOL newVal)

Interface ISegments3

AlternateSweep Property

Description Sets sweeps to either alternate or chopped.

VB Syntax *object*.AlternateSweep = *value*

Variable [\(Type\)](#) - Description

object Channel (**object**)

or

CalSet (**object**) - Read-only property

value (**boolean**) - Choose either:

False - Sweep mode set to **Chopped** - reflection and transmission are measured on the same sweep.

True - Sweep mode set to **Alternate** - reflection and transmission measured on separate sweeps. Improves Mixer bounce and Isolation measurements. Increases cycle time.

Return Type boolean

Default False (0)

Examples `chan.AlternateSweep = True 'Write`

`altSwp = chan.AlternateSweep 'Read`

C++ Syntax HRESULT AlternateSweep(VARIANT_BOOL *pVal)
HRESULT AlternateSweep(VARIANT_BOOL newVal)

Interface IChannel

ICalSet3

AmbientTemperature Property

Description Sets and returns the temperature at which the current noise measurement is occurring.
[Learn more.](#)

VB Syntax *noiseCal.AmbientTemperature = value*

Variable [\(Type\)](#) - Description

noiseCal A [NoiseCal](#) (**object**)

value (**double**) Ambient temperature in Kelvin.

Return Type Double

Default 295

Examples `noise.AmbientTemperature = 289` **Write**

`temp = noise.AmbientTemperature` **Read**

C++ Syntax HRESULT get_AmbientTemperature(Double* pValue)

HRESULT put_AmbientTemperature(Double pNewValue)

Interface INoiseCal

Last Modified:

6-Sep-2007 MX New topic

AnalysisCWFreq Property

Description Set and return the CW frequency for a compression analysis trace.

VB Syntax `gcaMeas.AnalysisCWFreq = value`

Variable [\(Type\)](#) - Description

gcaMeas A [GainCompressionMeas](#) (object)

value (**Double**) CW frequency in Hz. Choose a frequency within the range of the gain compression channel.

Return Type Double

Default Not Applicable

Examples `gcaMeas.AnalysisCWFreq = 1e9 'Write`

`cwfreq = gca.AnalysisCWFreq 'Read`

C++ Syntax HRESULT get_AnalysisCWFreq(Double* value)

HRESULT put_AnalysisCWFreq(Double value)

Interface IGainCompressionMeas

Last Modified:

3-Sep-2009 MX New topic

AnalysisEnable Property

Description Set and read the (ON | OFF) state of Gain Compression Analysis.

VB Syntax `gcaMeas.AnalysisEnable = value`

Variable [\(Type\)](#) - Description

`gcaMeas` A [GainCompressionMeas](#) (**object**)

`value` (**Boolean**) Choose from:

False - Disable GCA analysis trace.

True - Enable GCA analysis trace.

Return Type Boolean

Default False

Examples `gcaMeas.AnalysisEnable = True` 'Write

`analysis = gca.AnalysisEnable` 'Read

C++ Syntax HRESULT get_AnalysisEnable(VARIANT_BOOL* value)

HRESULT put_AnalysisEnable(VARIANT_BOOL value)

Interface IGainCompressionMeas

Last Modified:

3-Sep-2009 MX New topic

AnalysisIsDiscreteFreq Property

Description Sets and returns whether the CW frequency for the compression analysis trace can be set to only the discrete frequencies or provides interpolation.

VB Syntax `gcaMeas.AnalysisIsDiscreteFreq = value`

Variable [\(Type\)](#) - Description

gcaMeas A [GainCompressionMeas](#) (**object**)

value (**Boolean**) Choose from:

False - Interpolated data points.

True - Discrete data points only.

Return Type Boolean

Default False

Examples

```
gcaMeas.AnalysisIsDiscreteFreq = True 'Write  
isDisc = gca.AnalysisIsDiscreteFreq 'Read
```

C++ Syntax HRESULT get_AnalysisIsDiscreteFreq(VARIANT_BOOL* value)
HRESULT put_AnalysisIsDiscreteFreq(VARIANT_BOOL value)

Interface IGainCompressionMeas

Last Modified:

3-Sep-2009 MX New topic

AnalysisXAxis Property

Description Sets and returns the type of data to display on the x-axis of a compression analysis trace.

VB Syntax `gcaMeas.AnalysisXAxis = value`

Variable [\(Type\)](#) - Description

gcaMeas A [GainCompressionMeas](#) (object)

value (Enum as NAGCAAnalysisXAxis) Choose from:

naPsourceAsXAxis (0) - Power from the source.

naPinAsXAxis (1) - Input power to the DUT.

Return Type Enum

Default naPinAsXAxis (1)

Examples `gcaMeas.AnalysisXAxis = naPinAsXAxis 'Write`

`xAxis = gca.AnalysisXAxis 'Read`

C++ Syntax HRESULT get_AnalysisXAxis(tagNAGCAAnalysisXAxis* value)

HRESULT put_AnalysisXAxis(tagNAGCAAnalysisXAxis value)

Interface IGainCompressionMeas

Last Modified:

3-Sep-2009 MX New topic

Read-only

Application Property

Description Returns the name of the Analyzer making measurements on the channel.

VB Syntax *chan*.Application

Variable [\(Type\)](#) - Description

chan A [Channel](#) (object)

Return Type object

Default None

Examples `rfna = chan.Application 'returns the Analyzer name`

C++ Syntax HRESULT get_Application(IApplication** Application)

Interface IChannel

ArrangeWindows Property

Description Sets the arrangement of all the windows. Overlay, Stack2, Split3 and Quad4 will create windows.

To control the state of one window, use [app.WindowState](#).

VB Syntax `app.ArrangeWindows = value`

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

value (enum **NAWindowModes**) - Choose from:

0 - naTile

1 - naCascade

2 - naOverlay

3 - naStack2

4 - naSplit3

5 - naQuad4

Return Type Not Applicable

Default naTile

Examples `app.ArrangeWindow = naTile 'Write`

C++ Syntax HRESULT put_ArrangeWindows(tagNAWindowModes newVal)

Interface IApplication

AttenuatorMode Property

Description	Sets or returns the mode of operation of the attenuator control for the specified port number. This command is automatically set to Manual when an Attenuator value is set.
VB Syntax	<i>object</i> .AttenuatorMode(<i>portNum</i>) = <i>value</i>
Variable	(Type) - Description
<i>object</i>	Channel (object)
	or
	CalSet (object) - Read-only property
<i>portNum</i>	(long) - Port number of attenuator control to be changed.
<i>value</i>	(enum NAModes) - Choose from:
	0 - naAuto - Attenuator control set to automatic. The analyzer will set the attenuator control appropriately to deliver the specified power at the source.
	1 - naManual - Specify the attenuator setting using chan. Attenuator (which automatically sets AttenuatorMode = naManual).
Return Type	NAModes
Default	0 - Auto
Examples	<pre>chan.AttenuatorMode(1) = naAuto 'Write</pre>
	<pre>attn = chan.AttenuatorMode(1) 'Read</pre>
C++ Syntax	HRESULT get_AttenuatorMode(long port, tagNAModes* pVal) HRESULT put_AttenuatorMode(long port, tagNAModes newVal)
Interface	IChannel ICalSet3

Attenuator Property

Description Sets or returns the value of the source attenuator for the specified port number. Sending this command automatically sets [AttenuatorMode](#) to Manual.

VB Syntax *object*.Attenuator(*portNum*) = *value*

Variable [\(Type\)](#) - Description

object [Channel](#) (**object**)

or

[CalSet](#) (**object**) - Read-only property

portNum (**long integer**) - Port number of source attenuator to be changed.

value (**double**) - Attenuation value. The range of settable values depends on the PNA model. To determine the valid settings, do one of the following:

- See [PNA models and options](#) to see the range and step size for each model / option.
- To determine the maximum attenuator value use [MaximumSourceStepAttenuator](#).

If an invalid attenuation setting is entered, the PNA will select the next lower valid value. For example, if 19 is entered, then for an E8361A, 10 dB attenuation will be selected.

Return Type Double

Default 20 dB

Examples

```
chan.Attenuator(1) = 20 'Write
```

```
attn = chan.Attenuator(cnum) 'Read
```

C++ Syntax HRESULT get_Attenuator(long port, double *pVal)
HRESULT put_Attenuator(long port, double newVal)

Interface IChannel
ICalSet3

Last Modified:

28-Mar-2011 Fixed typo
25-Oct-2007 Edit value text
30-Apr-2007 Minor edits

AutoCWSweepTime Property

Description This command replaces [AutoIFBandWidth Property](#).

Sets and returns the state of automatic CW sweep time (used in Pulse Profile mode).

VB Syntax `pulseMeas.AutoCWSweepTime = bool`

Variable [\(Type\) - Description](#)

pulseMeas A [PulseMeasurementControl](#) (object)

bool **True** - In Pulse Profile mode, adjusts the default X-axis start time to zero and the stop time to double the Pulse Width. This allows you to see one complete pulse.

False - The Sweep Time is not changed.

Return Type Boolean

Default True

Examples

```
pulse.AutoCWSweepTime = True 'Write  
value = pulse.AutoCWSweepTime 'Read
```

C++ Syntax HRESULT get_AutoCWSweepTime(VARIANT_BOOL *pVal);
HRESULT put_AutoCWSweepTime(VARIANT_BOOL newVal);

Interface IPulseMeasurementControl2

Last modified:

9-May-2011 New topic

AutoDetection Property

Description Choose to automatically or manually set pulse mode (Narrowband or Wideband) for the channel.

VB Syntax `pulseMeas.AutoDetection = bool`

Variable [\(Type\)](#) - Description

pulseMeas A [PulseMeasurementControl](#) (object)

bool **False** - Manually set the pulse mode. Use [WideBandDectionState](#) to set the pulse mode.

True - Automatically set the pulse mode.

Return Type Boolean

Default True

Examples

```
pulse.AutoDetection = True 'Write  
value = pulse.AutoDetection 'Read
```

C++ Syntax HRESULT get_AutoDetection(VARIANT_BOOL *pVal);
HRESULT put_AutoDetection(VARIANT_BOOL newVal);

Interface IPulseMeasurementControl

Last Modified:

11-Mar-2010 New topic

AutoIFBandWidth Property - Superseded

Description This command is replaced by: [AutoCWSweepTime Property](#).

In Wideband pulse mode, choose to set the IF bandwidth automatically or manually.

VB Syntax `pulseMeas.AutoIFBandWidth = bool`

Variable [\(Type\)](#) - Description

pulseMeas A [PulseMeasurementControl](#) (**object**)

bool **False** - Manually set the IFBW for the measurement.

True - Automatically set the IFBW for the measurement.

Return Type Boolean

Default True

Examples `pulse.AutoIFBandWidth = True 'Write`

`value = pulse.AutoIFBandWidth 'Read`

C++ Syntax `HRESULT get_AutoIFBandWidth(VARIANT_BOOL *pVal);`

`HRESULT put_AutoIFBandWidth(VARIANT_BOOL newVal);`

Interface IPulseMeasurementControl

Last Modified:

May 9, 2011 Superseded

11-Mar-2010 New topic

AutoIFBWAdjustment Property

Description Set and read auto IFBW adjustment ON | OFF state for Gain Compression measurements.

VB Syntax `gca.AutoIFBWAdjustment = value`

Variable [\(Type\)](#) - Description

gca A [GainCompression](#) (object)

value (**Boolean**) - Auto IFBW adjustment state. Choose from:

False - Sets auto IFBW adjustment OFF

True - Sets auto IFBW adjustment ON

Return Type Boolean

Default ON

Examples `gca.AutoIFBWAdjustment = True 'Write`

`aifbw = gca.AutoIFBWAdjustment 'Read`

C++ Syntax HRESULT get_AutoIFBWAdjustment(VARIANT_BOOL* bState)

HRESULT put_AutoIFBWAdjustment(VARIANT_BOOL bState)

Interface IGainCompression

Last Modified:

8-Nov-2007 MX New topic

AutoIFGain Property

Type topic text here.

AutoOptimizePRF Property

Description In Narrowband pulse mode, choose to set the Pulse Repetition Frequency automatically or manually. This is labeled "Optimize Pulse Frequency" on the user-interface. To make changes manually, use [MasterFrequency Property](#) or [MasterPeriod Property](#).

VB Syntax `pulseMeas.AutoOptimizePRF = bool`

Variable [\(Type\)](#) - Description

pulseMeas A [PulseMeasurementControl](#) (object)

bool **False** - Manually set the PRF for the measurement.

True - Automatically set the PRF for the measurement.

Return Type Boolean

Default True

Examples

```
pulse.AutoOptimizePRF = True 'Write  
value = pulse.AutoOptimizePRF 'Read
```

C++ Syntax HRESULT get_AutoOptimizePRF(VARIANT_BOOL *pVal);
HRESULT put_AutoOptimizePRF(VARIANT_BOOL newVal);

Interface IPulseMeasurementControl

Last Modified:

11-Mar-2010 New topic

AutoOrient Property

Description Sets ECAL module automatic orientation ON or OFF.

VB Syntax *obj*.**AutoOrient** = *bool*

Variable [\(Type\)](#) - Description

obj [SMCType](#) (object)

or

[VMCType](#) (object)

bool (Boolean)

True - Set AutoOrientation ON

False - Set AutoOrientation OFF

Return Type Boolean

Default True

Examples `Smc.AutoOrient = True`

C++ Syntax HRESULT put_AutoOrient(VARIANT_BOOL bAutoOrient);

HRESULT get_AutoOrient(VARIANT_BOOL *bAutoOrient);

Interface SMCType

VMCType

AutoOrientTunerTuner Property

Description	Sets the state of auto orientation for a noise tuner during Noise Figure for NFX.
VB Syntax	<i>nfCal</i> . AutoOrientTuner = <i>bool</i>
Variable	(Type) - Description
<i>nfCal</i>	A NoiseCal (object)
<i>bool</i>	(Boolean)
	True - Set AutoOrientTuner ON
	False - Set AutoOrientTuner OFF
Return Type	Boolean
Default	True
Examples	<pre>nfCal.AutoOrientTuner = True</pre>
C++ Syntax	HRESULT put_AutoOrientTuner(VARIANT_BOOL bEnable); HRESULT get_AutoOrientTuner(VARIANT_BOOL *bEnable);
Interface	INoiseCal2

Last Modified:

27-Oct-2009 MX New topic

AutoPortExtConfig Property

Description Sets the frequency span that is used to calculate Automatic Port Extension. [Learn more about calculating Automatic Port Extension.](#)

VB Syntax `fixture.AutoPortExtConfig = value`

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

value (ENUM as NAAutoPortExtConfig)

0 naAPEC_CSPN - Use current span.

1 naAPEC_AMKR - Use active marker frequency.

2 naAPEC_USPN - Use custom user span. Use [AutoPortExtSearchStart Property](#) and [AutoPortExtSearchStop Property](#) to specify start and stop frequency.

Return Type ENUM

Default **0 naAPEC_CSPN**

Examples

```
fixture.AutoPortExtConfig = naAPEC_AMKR
```

```
value = fixture.AutoPortExtConfig 'Read
```

C++ Syntax HRESULT get_AutoPortExtConfig(tagNAAutoPortExtConfig *pVal);

HRESULT put_AutoPortExtConfig(tagNAAutoPortExtConfig Val);

Interface IFixturing2

AutoPortExtDCOffset Property

Description Specifies whether or not to include DC Offset as part of automatic port extension. Learn more about [Automatic DC Offset](#). Only allowed when [AutoPortExtLoss Property](#) is set to ON.

VB Syntax *fixture.AutoPortExtDCOffset = bool*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

bool **True** - Includes DC Offset correction.

False - Does NOT include DC Offset correction.

Return Type Boolean

Default False

Examples

```
fixture.AutoPortExtDCOffset = True
```

```
value = fixture.AutoPortExtDCOffset 'Read
```

C++ Syntax HRESULT get_AutoPortExtDCOffset(VARIANT_BOOL *pState);
HRESULT put_AutoPortExtDCOffset(VARIANT_BOOL bState);

Interface IFixturing2

AutoPortExtLoss Property

Description Specifies whether or not to include loss correction as part of automatic port extension.
[Learn more about Loss Compensation in port extension.](#)

VB Syntax *fixture*.AutoPortExtLoss = *bool*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

bool **True** - Includes Loss correction.

False - Does NOT include Loss correction.

Return Type Boolean

Default False

Examples

```
fixture.AutoPortExtLoss = True
```

```
value = fixture.AutoPortExtLoss 'Read
```

C++ Syntax HRESULT get_AutoPortExtLoss(VARIANT_BOOL *pState);
HRESULT put_AutoPortExtLoss(VARIANT_BOOL bState);

Interface IFixturing2

AutoPortExtSearchStart Property

Description Set the start frequency for custom user span. Only applies when [fixture.AutoPortExtConfig = 0 naAPEC_CSPN](#).

[Learn more about User Span.](#)

VB Syntax `fixture.AutoPortExtSearchStart = value`

Variable (Type) - Description

fixture A [Fixturing](#) (object)

value (Double) User span start value. Must be within the frequency range of the active channel and less than the value set by [AutoPortExtSearchStop Property](#)

Return Type Double

Default Start frequency of the current active channel.

Examples

```
fixture.AutoPortExtSearchStart = 1E9  
value = fixture.AutoPortExtSearchStart 'Read
```

C++ Syntax HRESULT get_AutoPortExtSearchStart(double *pdVal);

HRESULT put_AutoPortExtSearchStart(double dVal);

Interface IFixturing2

AutoPortExtSearchStop Property

Description Set the stop frequency for custom user span. Only applies when [fixture.AutoPortExtConfig = 0 naAPEC_CSPN](#).

[Learn more about User Span.](#)

Only applies when [fixture.AutoPortExtConfig = 0 naAPEC_CSPN](#)

VB Syntax `fixture.AutoPortExtSearchStop = value`

Variable (Type) - Description

fixture A [Fixturing \(object\)](#)

value (Double) User span stop value. Must be within the frequency range of the active channel and greater than the value set by [AutoPortExtSearchStart Property](#)

Return Type Double

Default Stop frequency of the current active channel.

Examples

```
fixture.AutoPortExtSearchStop = 1E9
```

```
value = fixture.AutoPortExtSearchStop 'Read
```

C++ Syntax HRESULT get_AutoPortExtSearchStop(double *pdVal);
HRESULT put_AutoPortExtSearchStop(double dVal);

Interface IFixturing2

AutoPortExtState Property

Description Enables and disables automatic port extensions on the specified port. All enabled ports will have their reference plane automatically adjusted after performing Automatic Port Extension.

VB Syntax *fixture.AutoPortExtState (port) = bool*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

port (**Integer**) Port number to enable or disable.

bool (Boolean)

True - Enables Auto Port Extensions

False - Disables Auto Port Extensions

Return Type Boolean

Default False

Examples `fixture.AutoPortExtState(1) = True`

`value = fixture.AutoPortExtState(2) 'Read`

C++ Syntax HRESULT get_AutoPortExtState(short port, VARIANT_BOOL *pState);

HRESULT put_AutoPortExtState(short port, VARIANT_BOOL bVal);

Interface IFixturing2

AutoPulseTiming Property

Description In Narrowband pulse mode, choose to set the delay and width automatically or manually. This is labeled "Autoselect Width and Delay" on the user-interface.

VB Syntax *pulseMeas.AutoPulseTiming = bool*

Variable [\(Type\)](#) - Description

pulseMeas A [PulseMeasurementControl](#) (object)

bool **False** - Manually set the delay and width for the measurement.

True - Automatically set the delay and width for the measurement.

Return Type Boolean

Default True

Examples

```
pulse.AutoPulseTiming = True 'Write  
value = pulse.AutoPulseTiming 'Read
```

C++ Syntax HRESULT get_AutoPulseTiming(VARIANT_BOOL *pVal);
HRESULT put_AutoPulseTiming(VARIANT_BOOL newVal);

Interface IPulseMeasurementControl

Last Modified:

11-Mar-2010 New topic

AutoSelectPulseGen Property

Description In Narrowband pulse mode, choose to set the pulse generator used to drive the source modulation automatically or manually.

VB Syntax *pulseMeas*.AutoSelectPulseGen = *bool*

Variable [\(Type\)](#) - Description

pulseMeas A [PulseMeasurementControl](#) (object)

bool **False** - Manually set source modulation drive for the measurement.

True - Automatically set source modulation drive for the measurement.

Return Type Boolean

Default True

Examples `pulse.AutoSelectPulseGen = True 'Write`

`value = pulse.AutoSelectPulseGen 'Read`

C++ Syntax HRESULT get_AutoSelectPulseGen(VARIANT_BOOL *pVal);

HRESULT put_AutoSelectPulseGen(VARIANT_BOOL newVal);

Interface IPulseMeasurementControl

Last Modified:

11-Mar-2010 New topic

AuxiliaryTriggerCount Property

Description Returns the number of aux trigger input / output connector pairs in the PNA

VB Syntax *value* = *app*.**AuxiliaryTriggerCount**

Variable [\(Type\)](#) - Description

value **(Long Integer)** Variable to store the returned value.

2 = PNA-X models

1 = All other PNA models

app An [Application](#) (**object**)

Return Type Long Integer

Default Not Applicable

Examples `ioConns = app.AuxiliaryTriggerCount`

C++ Syntax HRESULT AuxiliaryTriggerCount(long *count);

Interface IApplication11

Last Modified:

6-Apr-2009 Replaced N5242A with PNA-X

14-Dec-2006 MX New topic

AuxTriggerScopeIsGlobal Property

Description Sets the Trigger OUT behavior to either Global or Channel. [Learn more about this setting.](#)

This command will cause the PNA to [Preset](#).

This setting remains until changed again using this command, or until the hard drive is changed or reformatted.

See the [AuxTrigger Object](#).

VB Syntax `pref.AuxTriggerScopeIsGlobal = value`

Variable [\(Type\)](#) - Description

pref A [Preferences](#) (object)

value (Boolean) - Choose from:

- **True** - Trigger properties apply to ALL channels (Global).
 - Default setting for **E836x and PNA-L** models.
 - Allows use of `command` to configure the external trigger properties.
 - "Per Point" trigger property is not settable. Use the channel's [Point trigger](#) setting.
- **False** - External Trigger properties apply to each channel independently.
 - Default setting for **PNA-X** models.
 - Must use [AuxTrigger](#) commands to configure the external trigger properties. [ExternalTriggerConnectionBehavior Property](#) will NOT work.
 - "Per Point" trigger output property is set using the channel's [Point trigger](#) setting **AND** [TriggerOutInterval Property](#).

Return Type Boolean

Default **True** - E836xB and PNA-L models

False - PNA-X models

Examples

```
pref.AuxTriggerScopeIsGlobal = 1 'Write
```

```
auxTrigPref = pref.AuxTriggerScopeIsGlobal 'Read
```

C++ Syntax HRESULT get_AuxTriggerScopeIsGlobal(VARIANT_BOOL * pref);

HRESULT put_AuxTriggerScopeIsGlobal(VARIANT_BOOL pref);

Interface IPreferences5

Last modified:

25-Feb-2008 Clarification

Jan 3, 2007 MX New command

AvailableMeasurementClasses Property

Description Returns a list of available measurement classes on the PNA.

VB Syntax *value* = *cap*.**AvailableMeasurementClasses**

Variable [\(Type\)](#) - Description

value (Variant) - Variable to store the returned list of measurement classes.

cap A [Capabilities](#) (object)

Return Type Variant

Default Not Applicable

Examples

```
'Read all measurement classes
Set app = CreateObject("AgilentPNA835x.Application")
Set cap = app.Capabilities
meas=cap.AvailableMeasurementClasses
dim i
For i = 0 To UBound(meas)
    msg = msg & meas(i) & vbCrLf
Next
MsgBox msg
```

C++ Syntax HRESULT get_AvailableMeasurementClasses(Variant *value);

Interface ICapabilities7

Last Modified:

23-May-2011 Added example

4-Nov-2010 MX New topic

AverageMode Property

Description Specifies the type of averaging to perform: Point or Sweep.

VB Syntax *chan.AverageMode = value*

Variable [\(Type\)](#) - Description

chan A [Channel](#) (object)

value (Enum as **naAverageMode**) - Average Type. Choose from:

0 - naPoint Averaging measurements are made on each data point before stepping to the next data point. (Not available on 'C' models).

1 - naSweep Averaging measurements are made on subsequent sweeps until the required number of averaging sweeps are performed.

Return Type Enum

Default 1- **naSweep**

Examples

```
chan.AverageMode = naSweep 'Write
```

```
avgType = chan.AverageMode ' Read
```

C++ Syntax HRESULT get_AverageMode(NAAverageMode * mode);

HRESULT put_AverageMode(NAAverageMode mode);

Interface IChannel16

Last Modified:

8-Jun-2009 Not available on 'C' models.

13-Oct-2008 MX New topic

AveragingCount Property

Description	Returns the number of sweeps that have been acquired and averaged into the measurements on this channel. AveragingFactor specifies the number of sweeps to average. AveragingCount indicates the progress toward that goal.
VB Syntax	<i>value</i> = <i>chan.AveragingCount</i>
Variable	(Type) - Description
<i>chan</i>	A Channel (object)
<i>value</i>	(Long Integer) - Variable to store the returned count
Return Type	Long Integer
Default	Not Applicable
Example	<pre>avgcount = chan.AveragingCount</pre>
C++ Syntax	HRESULT get_AveragingCount(long* count)
Interface	IChannel

AveragingFactor Property

Description Specifies the number of measurements to combine for an average. Must also turn averaging ON by setting *chan.Averaging* = 1.

VB Syntax *chan.AveragingFactor* = *value*

Variable [\(Type\)](#) - Description

chan A Channel (**object**)

value (**Long Integer**) - Number of measurement sweeps to average. Choose any number between **1** and **65536** (2¹⁶).

Return Type Long Integer

Default 1

Examples

```
chan.AveragingFactor = 5 'Write
avgfact = chan.AveragingFactor ' Read
```

C++ Syntax HRESULT get_AveragingFactor(long *pVal)
HRESULT put_AveragingFactor(long newVal)

Interface IChannel

Last Modified:

16-Apr-2009 Updated for Point Averaging

Averaging Property

Description Turns trace averaging ON or OFF for all measurements on the channel. Averaging is only allowed on ratioed measurements; not on single input measurements.

VB Syntax `chan.Averaging = state`

Variable [\(Type\)](#) - Description

chan A [Channel](#) (**object**)

state (**boolean**)
False - Turns averaging OFF
True - Turns averaging ON

Return Type Boolean

Default False

Examples

```
chan.Average = True 'Write
```

```
averg = chan.Averaging 'Read
```

C++ Syntax HRESULT get_Averaging(BOOL *pVal)
HRESULT put_Averaging(BOOL newVal)

Interface IChannel

AvoidSpurs Property

Description	Sets and returns the state of the avoid spurs feature.
VB Syntax	<i>mixer</i> . AvoidSpurs = <i>boolean</i>
Variable	(Type) - Description
<i>mixer</i>	A Mixer (object)
	A Converter (object)
<i>value</i>	(Boolean) - State of avoid spurs feature. Choose from
	False Avoid spurs OFF
	True Avoid spurs ON
Return Type	Boolean
Default	False
Examples	<pre>conv.AvoidSpurs = True 'Write</pre>
	<pre>variable = conv.AvoidSpurs 'Read</pre>
C++ Syntax	HRESULT get_AvoidSpurs(Bool *bVal)
	HRESULT put_AvoidSpurs(Bool newVal)
Interface	IMixer3
	IConverter5

Last Modified:

25-Jan-2011 Added converter object

Background Property

Description Set and return the background color for the PNA display or hardcopy print.

VB Syntax *colors*.**Background** = *value*

Variable [\(Type\)](#) - Description

colors A [ComColors](#) (object)

value (**Long Integer**) - RGB color of the Background pen.

Convert the three RGB colors to an integer as follows:

```
RGB = R+(G*2^8)+(B*2^16)
```

To find the three RGB values from the [Display Colors dialog](#), click **Change Color**, then **Define Custom Color**.

Return Type Long

Default Display = 0,0,0 (Black)

Print = 255,255,255 (White)

Examples

```
R = 10
G = 10
B = 10
RGB = R+(G*2^8)+(B*2^16)
colors.Background = RGB 'Write
color = colors.Background 'Read
```

C++ Syntax HRESULT get_Background(long* pVal);

HRESULT put_Background(long newVal);

Interface IComColors

Last Modified:

7-Aug-2009 MX New topic

BackOff Property

Description Sets and returns the backoff value used to calculate various PNOP parameters.

Also set [PinOffset Property](#).

A sweep must be executed (single or continuous) and [SearchPowerNormalOperatingPoint Method](#) must be sent before reading marker results.

To turn off the PNOP markers, either turn them off individually or [DeleteAllMarkers](#).

To search a [User Range](#) with the PNOP search, first activate marker 1. The user range used with the PNOP search only applies to marker 1 searching for the linear gain value. The other markers may fall outside the user range.

VB Syntax *pnop*.**BackOff** = *value*

Variable [\(Type\)](#) - Description

pnop A [PNOP](#) (object)

value (**double**) - Backoff value in dB. Choose any number between:-**500** and **500**

Return Type Double

Default 0 dB

Examples `backoff = pnop.BackOff 'Read`

[See example program](#)

C++ Syntax HRESULT put_BackOff(double newVal);

HRESULT get_BackOff(double* pNewVal)

Interface IPNOP

Last Modified:

19-Feb-2010 MX New topic

BackOffGain Property

Description Returns the BackOffGain result of the PNOP marker search.

$PBO\ Gain = PBO\ Out - PBO\ In$

VB Syntax *bOffGain* = *pnop*.**BackOffGain**

Variable [\(Type\)](#) - [Description](#)

bOffGain (**double**) - Variable to store returned value

pnop A [PNOP](#) (**object**)

Return Type Double

Default Not applicable

Examples `bOffGain = pnop.BackOffGain 'Read`

[See example program](#)

C++ Syntax HRESULT get_BackOffGain(double* pNewVal)

Interface IPNOP

Last Modified:

19-Feb-2010 MX New topic

BackOffPin Property

Description Returns the BackOffPin result of the PNOP marker search.

PBO In = Marker 2 X-axis

VB Syntax *bOffPin* = *pnop*.**BackOffPin**

Variable [\(Type\)](#) - **Description**

bOffPin (**double**) - Variable to store returned value

pnop A [PNOP](#) (**object**)

Return Type Double

Default Not applicable

Examples `bOffPin = pnop.BackOffPin 'Read`

[See example program](#)

C++ Syntax HRESULT get_BackOffPin(double* pNewVal)

Interface IPNOP

Last Modified:

19-Feb-2010 MX New topic

BackOffPout Property

Description Returns the BackOffPout result of the PNOP marker search.

PBO Out = Marker 2 Y-axis

VB Syntax *bOffPout* = *pnop*.**BackOffPout**

Variable [\(Type\)](#) - [Description](#)

bOffPout (**double**) - Variable to store returned value

pnop A [PNOP](#) (**object**)

Return Type Double

Default Not applicable

Examples `bOffPout = pnop.BackOffPout 'Read`

[See example program](#)

C++ Syntax HRESULT get_BackOffPout(double* pNewVal)

Interface IPNOP

Last Modified:

19-Feb-2010 MX New topic

BalancedMode Property

Description	Sets and returns whether the balanced transform is ON or OFF
VB Syntax	<i>balMeas</i> . BalancedMode = <i>value</i>
Variable	(Type) - Description
<i>balMeas</i>	A BalancedMeasurement (object)
<i>value</i>	(Boolean) - State of balanced transform. Choose from False Balanced Transform OFF True Balanced Transform ON
Return Type	Boolean
Default	False
Examples	<pre>balMeas.BalancedMode = True 'Write</pre> <pre>variable = balMeas.BalancedMode 'Read</pre>
C++ Syntax	HRESULT get_BalancedMode(VARIANT_BOOL *bVal) HRESULT put_BalancedMode(VARIANT_BOOL newVal)
Interface	IBalancedMeasurement

BalPort1PhaseOffset Property

Description Sets and returns the phase offset between the two ports that comprise Balanced port 1. [balStim.Mode](#) must be set to a True Stimulus mode. Applicable only with [Opt 460 - iTMSA](#).

VB Syntax *balStim.BalPort1PhaseOffset* = *value*

Variable [\(Type\)](#) - Description

balStim A [BalancedStimulus](#) (object)

value (**Double**) - Phase Offset in degrees. Choose a value between -360 and 360.

Return Type Double

Default 0

Examples `balStim.BalPort1PhaseOffset = 10 'Write`

`variable = balStim.BalPort1PhaseOffset 'Read`

C++ Syntax HRESULT get_BalPort1PhaseOffset (double *pVal)

HRESULT put_BalPort1PhaseOffset (double newVal)

Interface IBalancedStimulus

Last Modified:

15-May-2008 MX New topic

BalPort1PowerOffset Property

Last Modified:

15-May-2008 MX New topic

BalPort1StartPhase Property

Description Sets and returns the start phase of a phase sweep.

VB Syntax *balStim*.**BalPort1StartPhase** = *value*

Variable [\(Type\)](#) - Description

balStim A [BalancedStimulus](#) (object)

value (**Double**) - Start phase in degrees. Choose a value between -360 and 360.

Return Type Double

Default 0

Examples `balStim.BalPort1StartPhase = 10 'Write`

`variable = balStim.BalPort1StartPhase 'Read`

C++ Syntax HRESULT get_BalPort1StartPhase (double *pVal)

HRESULT put_BalPort1StartPhase (double newVal)

Interface IBalancedStimulus2

Last Modified:

3-Mar-2009 MX New topic (A.08.50)

BalPort1StopPhase Property

Description Sets and returns the stop phase of a phase sweep.

VB Syntax *balStim.BalPort1StopPhase = value*

Variable [\(Type\)](#) - Description

balStim A [BalancedStimulus](#) (object)

value (**Double**) - Stop phase in degrees. Choose a value between -360 and 360.

Return Type Double

Default 0

Examples `balStim.BalPort1StopPhase = 10` 'Write

`variable = balStim.BalPort1StopPhase` 'Read

C++ Syntax HRESULT get_BalPort1StopPhase (double *pVal)

HRESULT put_BalPort1StopPhase (double newVal)

Interface IBalancedStimulus2

Last Modified:

3-Mar-2009 MX New topic (A.08.50)

BalPort2PhaseOffset Property

Description Sets and returns the phase offset between the two ports that comprise Balanced port 2. [balStim.Mode](#) must be set to a True Stimulus mode. Applicable only with [Opt 460 - iTMSA](#).

VB Syntax *balStim.BalPort2PhaseOffset* = *value*

Variable [\(Type\)](#) - Description

balStim A [BalancedStimulus](#) (object)

value (**Double**) - Phase Offset in degrees. Choose a value between -360 and 360.

Return Type Double

Default 0

Examples `balStim.BalPort2PhaseOffset = 10 'Write`

`variable = balStim.BalPort2PhaseOffset 'Read`

C++ Syntax HRESULT get_BalPort2PhaseOffset (double *pVal)

HRESULT put_BalPort2PhaseOffset (double newVal)

Interface IBalancedStimulus

Last Modified:

15-May-2008 MX New topic

BalPort2PowerOffset Property

Description Sets and returns the power offset between the two ports that comprise Balanced port 2. [balStim.Mode](#) must be set to a True Stimulus mode. Applicable only with [Opt 460 - iTMSA](#).

VB Syntax *balStim.BalPort2PowerOffset = value*

Variable [\(Type\)](#) - Description

balStim A [BalancedStimulus](#) (object)

value (**Double**) - Power Offset in dB. Choose a value between .

Return Type Double

Default 0

Examples `balStim.BalPort2PowerOffset = 2 'Write`

`variable = balStim.BalPort2PowerOffset 'Read`

C++ Syntax HRESULT get_BalPort2PowerOffset (double *pVal)

HRESULT put_BalPort2PowerOffset (double newVal)

Interface IBalancedStimulus

Last Modified:

15-May-2008 MX New topic

BalPort2StartPhase Property

Description	Sets and returns the start phase of a phase sweep.
VB Syntax	<i>balStim</i> . BalPort2StartPhase = <i>value</i>
Variable	(Type) - Description
<i>balStim</i>	A BalancedStimulus (object)
<i>value</i>	(Double) - Start phase in degrees. Choose a value between -360 and 360.
Return Type	Double
Default	0
Examples	<pre>balStim.BalPort2StartPhase = 10 'Write</pre> <pre>variable = balStim.BalPort2StartPhase 'Read</pre>
C++ Syntax	HRESULT get_BalPort2StartPhase (double *pVal) HRESULT put_BalPort2StartPhase (double newVal)
Interface	IBalancedStimulus2

Last Modified:

3-Mar-2009 MX New topic (A.08.50)

BalPort2StopPhase Property

Description Sets and returns the stop phase of a phase sweep.

VB Syntax *balStim.BalPort2StopPhase = value*

Variable [\(Type\)](#) - Description

balStim A [BalancedStimulus](#) (object)

value (**Double**) - Stop phase in degrees. Choose a value between -360 and 360.

Return Type Double

Default 0

Examples `balStim.BalPort2StopPhase = 10` 'Write

`variable = balStim.BalPort2StopPhase` 'Read

C++ Syntax HRESULT get_BalPort2StopPhase (double *pVal)

HRESULT put_BalPort2StopPhase (double newVal)

Interface IBalancedStimulus2

Last Modified:

3-Mar-2009 MX New topic (A.08.50)

BandwidthTarget Property

Description Sets the insertion loss value at which the bandwidth of a filter is measured (using [BandwidthTracking](#) or [SearchFilterBandwidth](#)). For example, if you want to determine the filter bandwidth 3 db below the bandpass peak value, set BandwidthTarget to **-3**.

VB Syntax *meas*.BandwidthTarget = *value*

Variable [\(Type\)](#) - Description

meas A Measurement (**object**)

value (**single**) - Target value. Choose any number between **-500** and **500**

Return Type Single

Default -3

Examples `meas.BandwidthTarget = -3 'Write`

`fbw = meas.BandwidthTarget 'Read`

C++ Syntax HRESULT put_BandwidthTarget(float target)
HRESULT get_BandwidthTarget(float* target)

Interface IMeasurement

BandwidthTracking Property

Description Searches continually (every sweep) for the current [BandwidthTarget](#) (default is -3). To search the filter bandwidth for ONE SWEEP only (not continually), use `meas.SearchFilterBandwidth`.

This feature uses markers 1-4. To turn off these markers, either turn them off individually or [DeleteAllMarkers](#).

The bandwidth statistics are displayed on the analyzer screen. To get the bandwidth statistics, use either [GetFilterStatistics](#) or [FilterBW](#), [FilterCF](#), [FilterLoss](#), or [FilterQ](#).

The analyzer screen will show either Bandwidth statistics OR Trace statistics; not both.

To restrict the search to a [UserRange](#) with the bandwidth search, first activate marker 1 and set the desired UserRange. Then send the SearchFilterBandwidth command. The user range used with bandwidth search only applies to marker 1 searching for the max value. The other markers may fall outside the user range.

VB Syntax `meas.BandwidthTracking = value`

Variable [\(Type\)](#) - Description

meas A [Measurement](#) (object)

value (boolean)

True - Turns bandwidth tracking ON

False - Turns bandwidth tracking OFF

Return Type Boolean

Default **False**

Examples `meas.BandwidthTracking = False 'Write`

`bwtrack = meas.BandwidthTracking 'Read`

C++ Syntax HRESULT put_BandwidthTracking(VARIANT_BOOL state)
HRESULT get_BandwidthTracking(VARIANT_BOOL* state)

Interface IMeasurement

BB_BalPort1Negative Property

Description With a Balanced - Balanced topology, returns the PNA port number that is connected to the Negative side of the DUT's logical Port 1.

Use [SetBBPorts Method](#) to set the port mapping for a Balanced - Balanced topology.

VB Syntax `var = balTopology.BB_BalPort1Negative`

Variable [\(Type\)](#) - Description

`var` (Long Integer) Variable to store the returned value.

`balTopology` A [BalancedTopology](#) (object)

Return Type Long Integer

Default Not Applicable

Examples `variable = balTop.BB_BalPort1Negative` 'Read

C++ Syntax HRESULT get_BB_BalPort1Negative(long *bVal)

Interface IBalancedTopology

BB_BalPort1Positive Property

Description With a Balanced - Balanced topology, returns the PNA port number that is connected to the Positive side of the DUT's logical Port 1.

Use [SetBBPorts Method](#) to set the port mapping for a Balanced - Balanced topology.

VB Syntax `var = balTopology.BB_BalPort1Positive`

Variable [\(Type\)](#) - Description

`var` (Long Integer) Variable to store the returned value.

`balTopology` A [BalancedTopology](#) (object)

Return Type Long Integer

Default Not Applicable

Examples `variable = balTop.BB_BalPort1Positive` 'Read

C++ Syntax HRESULT get_BB_BalPort1Positive(long *bVal)

Interface IBalancedTopology

BB_BalPort2Negative Property

Description With a Balanced - Balanced topology, returns the PNA port number that is connected to the Negative side of the DUT's logical Port 2.

Use [SetBBPorts Method](#) to set the port mapping for a Balanced - Balanced topology.

VB Syntax `var = balTopology.BB_BalPort2Negative`

Variable [\(Type\)](#) - Description

`var` (Long Integer) Variable to store the returned value.

`balTopology` A [BalancedTopology](#) (object)

Return Type Long Integer

Default Not Applicable

Examples `variable = balTop.BB_BalPort2Negative` 'Read

C++ Syntax HRESULT get_BB_BalPort2Negative(long *bVal)

Interface IBalancedTopology

BB_BalPort2Positive Property

Description With a Balanced - Balanced topology, returns the PNA port number that is connected to the Positive side of the DUT's logical Port 2.

Use [SetBBPorts Method](#) to set the port mapping for a Balanced - Balanced topology.

VB Syntax `var = balTopology.BB_BalPort2Positive`

Variable [\(Type\)](#) - Description

`var` (Long Integer) Variable to store the returned value.

`balTopology` A [BalancedTopology](#) (object)

Return Type Long Integer

Default Not Applicable

Examples `variable = balTop.BB_BalPort2Positive` **'Read**

C++ Syntax HRESULT get_BB_BalPort2Positive(long *bVal)

Interface IBalancedTopology

BalSMeasurement Property

Description Sets and returns the measurement for the Balanced - Single-ended topology.

VB Syntax *balMeas*.BalSMeasurement = *value*

Variable [\(Type\)](#) - Description

balMeas A [BalancedMeasurement](#) (object)

value (**String**) - Balanced - Single-ended measurement parameter. Not case sensitive. Choose from:

Sdd11	Sdc11	Sds12
Scd11	Sc11	Scs12
Ssd21	Ssc21	Sss22
Imb	CMRR1 (Ssd21/Ssc21)	CMRR2 (Sds12/Scs12)

Return Type String

Default Sdd11

Examples

```
balMeas.BalSMeasurement = "Sdd11" 'Write
```

```
variable = balMeas.BalSMeasurement 'Read
```

C++ Syntax HRESULT get_BalSMeasurement(BSTR *pVal)

HRESULT put_BalSMeasurement(BSTR newVal)

Interface IBalancedMeasurement3

Last Modified:

Unknown MX New topic

BBalMeasurement Property

Description Sets and returns the measurement for the Balanced - Balanced topology.

VB Syntax `balMeas.BBalMeasurement = value`

Variable [\(Type\)](#) - Description

`balMeas` A [BalancedMeasurement](#) (object)

`value` (**String**) - Balanced - Balanced Measurement parameter. Not case sensitive.
Choose from:

Sdd11	Sdd12	Sdc11	Sdc12
Sdd21	Sdd22	Sdc21	Sdc22
Scd11	Scd12	Sc11	Sc12
Scd21	Scd22	Sc21	Sc22
Imb1	Imb2	CMRR -(Sdd21/Sc11)	

Return Type String

Default Sdd11

Examples `balMeas.BBalMeasurement = "Sdd11" 'Write`

`variable = balMeas.BBalMeasurement 'Read`

C++ Syntax HRESULT get_BBalMeasurement(BSTR *pVal)

HRESULT put_BBalMeasurement(BSTR newVal)

Interface IBalancedMeasurement

Last Modified:

Unknown MX New topic

BeginResponse Property

Description When constructing a limit line, specifies the amplitude value of the start of a limit segment.

VB Syntax *limtseg*.BeginResponse = *value*

Variable [\(Type\)](#) - Description

limtseg A LimitSegment (**object**)

value (**double**) - Amplitude value. No units

Return Type Double

Default 0

Examples

```
Set limtseg = meas.LimitTest(1)
limtseg.BeginResponse = 10 'Write
```

```
BegResp = limtseg.BeginResponse 'Read
```

C++ Syntax HRESULT get_BeginResponse(double *pVal)
HRESULT put_BeginResponse(double newVal)

Interface ILimitSegment

BeginStimulus Property

Description When constructing a limit line, specifies the beginning X-axis value.

VB Syntax *limtseg*.BeginStimulus = *value*

Variable [\(Type\)](#) - Description

limtseg A LimitSegment (**object**)

value (**double**) - Stimulus value. No units

Return Type Double

Default 0

Examples

```
Set limtseg = meas.LimitTest(1)
limtseg.Type = naLimitSegmentType_Maximum
limtseg.BeginStimulus = 3e9
limtseg.EndStimulus = 4e9
limtseg.BeginResponse = 10
limtseg.EndResponse = 10
```

```
BegStim = limtseg.BeginStimulus 'Read
```

C++ Syntax HRESULT get_BeginStimulus(double *pVal)
HRESULT put_BeginStimulus(double newVal)

Interface ILimitSegment

BroadbandTuningSpan Property

Description Sets and returns the frequency span for the broadband tuning sweep.

VB Syntax *obj*.**BroadbandTuningSpan** = *value*

Variable [\(Type\)](#) - Description

obj An [EmbeddedLO](#) (**object**) or

A [ConverterEmbeddedLO](#) (**object**)

value (**Double**) Broadband frequency span in Hz.

Return Type (**Double**)

Default 3 MHz

Examples

```
embedLO.BroadbandTuningSpan = 1E6 'write
```

```
value = embedLO.BroadbandTuningSpan 'read
```

C++ Syntax HRESULT get_BroadbandTuningSpan(double* span);

HRESULT put_BroadbandTuningSpan(double span);

Interface IEmbeddedLO

Last Modified:

12-Aug-2009 Added ConvEmbedLO object

18-Apr-2007 MX New topic

BS_BalPortNegative Property

Description With a Balanced - Single-ended topology, returns the PNA port number that is connected to the Negative side of the DUT's balanced port.

Use [SetSBPorts Method](#) to set the port mapping for a Balanced - Single-ended topology.

VB Syntax *var* = *balTopology*.**BS_BalPortNegative**

Variable [\(Type\)](#) - Description

var (Long Integer) Variable to store the returned value.

balTopology A [BalancedTopology](#) (object)

Return Type Long Integer

Default Not Applicable

Examples `variable = balTop.BS_BalPortNegative 'Read`

C++ Syntax HRESULT get_BS_BalPortNegative(long *bVal)

Interface IBalancedTopology2

Last modified:

25-Apr-2012 New topic

BS_BalPortPositive Property

Description With a Balanced - Single-ended topology, returns the PNA port number that is connected to the Positive side of the DUT's balanced port.

Use [SetSBPorts Method](#) to set the port mapping for a Balanced - Single-ended topology.

VB Syntax `var = balTopology.BS_BalPortPositive`

Variable [\(Type\)](#) - Description

`var` (Long Integer) Variable to store the returned value.

`balTopology` A [BalancedTopology](#) (object)

Return Type Long Integer

Default Not Applicable

Examples `variable = balTop.BS_BalPortPositive 'Read`

C++ Syntax HRESULT get_BS_BalPortPositive(long *bVal)

Interface IBalancedTopology2

Last modified:

25-Apr-2012 New topic

BS_SEPort Property

Description With a Balanced - Single-ended topology, returns the PNA port number that is connected to the Single-ended port.

Use [SetSBPorts Method](#) to set the port mapping for a Balanced - Single-ended topology.

VB Syntax *var* = *balTopology*.**BS_SEPort**

Variable [\(Type\)](#) - Description

var (Long Integer) Variable to store the returned value.

balTopology A [BalancedTopology](#) (object)

Return Type Long Integer

Default Not Applicable

Examples `variable = balTop.BS_SEPort` **'Read'**

C++ Syntax HRESULT get_BS_SEPort(long *bVal)

Interface IBalancedTopology2

Last modified:

25-Apr-2012 New topic

BucketNumber Property

Description	Sets or returns the bucket number (data point) for the active marker. When the markers are interpolated (non-discrete) , the returned value is the nearest marker bucket position.
VB Syntax	<i>mark</i> . BucketNumber = <i>value</i>
Variable	(Type) - Description
<i>mark</i>	A Marker (object)
<i>value</i>	(long integer) - Data point. Choose any number between 0 and the measurement's number of data points - 1. For example, with Number of points = 201, choose between 0 and 200
Return Type	Long Integer
Default	The first marker is set to the middle of the span. Subsequent markers are set to the bucket number of the previously active marker.
Examples	<pre>mark.BucketNumber = 100 'moves the active marker to data point 100 -Write pointNumber = mark.BucketNumber 'returns the data point number of the marker object. When the markers are interpolated (non-discrete), the returned value is the nearest marker bucket position.</pre>
C++ Syntax	HRESULT get_BucketNumber(long *pVal) HRESULT put_BucketNumber(long newVal)
Interface	IMarker

C0 Property

Description Sets and Returns the C0 (C-zero) value (the first capacitance value) for the calibration standard.

To set the other capacitance values, use [C1](#), [C2](#), [C3](#)

VB Syntax `calstd.C0 = value`

Variable [\(Type\)](#) - Description

calstd A CalStandard (**object**). Use calKit.[GetCalStandard](#) to get a handle to the standard.

value (**single**) - Value for C0 in femtofarads (1E-15)

Return Type Single

Default Not Applicable

Examples `calstd.C0 = 15 'Write the value of C0 to 15femtofarads`

`cap0 = calstd.C0 'Read the value of C0`

C++ Syntax HRESULT get_C0(float *pVal)
HRESULT put_C0(float newVal)

Interface ICalStandard

C1 Property

Description Sets and Returns the C1 value (the second capacitance value) for the calibration standard.

To set the other capacitance values, use [C0](#), [C2](#), [C3](#).

VB Syntax *calstd.C1 = value*

Variable [\(Type\)](#) - **Description**

calstd A CalStandard (**object**). Use calKit.[GetCalStandard](#) to get a handle to the standard.

value (**single**) - Value for C1.

Return Type Single

Default Not Applicable

Examples `calstd.C1 = 15 'Write the value of C1.`

`cap1 = calstd.C1 'Read the value of C1.`

C++ Syntax HRESULT get_C1(float *pVal)
HRESULT put_C1(float newVal)

Interface ICalStandard

C2 Property

Description Sets and Returns the C2 value (the third capacitance value) for the calibration standard.

To set the other capacitance values, use [C0](#), [C1](#), [C3](#).

VB Syntax `calstd.C2 = value`

Variable [\(Type\)](#) - Description

calstd A CalStandard (**object**). Use calKit.[GetCalStandard](#) to get a handle to the standard.

value (**single**) - Value for C2.

Return Type Single

Default Not Applicable

Examples

```
calstd.C2 = 15 'Write the value of C2.
```

```
cap2 = calstd.C2 'Read the value of C2
```

C++ Syntax HRESULT get_C2(float *pVal)
HRESULT put_C2(float newVal)

Interface ICalStandard

C3 Property

Description Sets and Returns the C3 value (the fourth capacitance value) for the calibration standard.

To set the other capacitance values, use [C0](#), [C1](#), [C2](#)

VB Syntax `calstd.C3 = value`

Variable [\(Type\)](#) - Description

calstd A CalStandard (**object**). Use calKit.[GetCalStandard](#) to get a handle to the standard.

value (**single**) - Value for C3.

Return Type Single

Default Not Applicable

Examples `calstd.C3 = 15 'Write the value of C3.`

`cap3 = calstd.C3 'Read the value of C3`

C++ Syntax HRESULT get_C3(float *pVal)
HRESULT put_C3(float newVal)

Interface ICalStandard

CalFactor Property

Description	Sets or returns the cal factor value associated with a power sensor cal factor segment.
VB Syntax	<i>calFactSeg</i> . CalFactor = <i>value</i>
Variable	(Type) - Description
<i>calFactSeg</i>	A PowerSensorCalFactorSegment (Object) or A PowerSensorCalFactorSegmentPMAR (Object)
<i>value</i>	(double) – Cal factor in percent. Choose any value between 1 and 150
Return Type	Double
Default	0
Examples	<pre>calFactSeg.CalFactor = 98 'Write factor = calFactSeg.CalFactor 'Read</pre>
C++ Syntax	HRESULT put_CalFactor(Double newVal); HRESULT get_CalFactor(Double *pVal);
Interface	IPowerSensorCalFactorSegment IPowerSensorCalFactorSegmentPMAR

Last Modified:

27-Aug-2009 MX New topic

CalibrationType Property **Superseded**

Description **Note:** This command has been replaced by [CalibrationTypeID_property](#), which provides selection of Calibration Type by string.

Specifies the type of calibration to perform or apply to the active S-Parameter measurement. This command determine the ports involved in the CalType by the ports being used by the active measurement.

For example:

- If the measurement is an S23, it uses ports 2 and 3.
- If the measurement is an S22 it will use the legacy load port to figure out which two ports form the caltype. The legacy load port is set using [CreateMeasurement](#).
- If **naCalType_ThreePort_SOLT** is specified on a 4-port PNA, an E_NA_DEPRECATED_COMMAND error is returned. There is no way to determine the intended three ports.
- If **naCalType_FourPort_SOLT** is specified on a 4-port PNA, it is obvious that the ports involved are ports 1,2,3, and 4.

Note: For FCA measurements, use [CalibrationName](#) and [CalibrationTypeID](#).

VB Syntax *meas.CalibrationType = type*

Variable [\(Type\)](#) - Description

meas A [Measurement](#) (object)

type (enum NACalType) - Calibration type. Choose from:

0 - naCalType_Response_Open

1 - naCalType_Response_Short

2 - naCalType_Response_Thru

3 - naCalType_Response_Thru_And_Isol

4 - naCalType_OnePort

5 - naCalType_TwoPort_SOLT

6 - naCalType_TwoPort_TRL

7 - naCalType_None

8 - naCalType_ThreePort_SOLT

9 - Custom

10 - naCalType_FourPort_SOLT

**Return
Type** NACalType

Default naCalType_None

Examples `meas.CalibrationType = naCalType_Response_Open 'Write`

`meascal = meas.CalibrationType 'Read`

C++ Syntax HRESULT put_CalibrationType (tagNACalType CalType)
HRESULT get_CalibrationType (tagNACalType* pCalType)

Interface IMeasurement

Read only

CalibrationName Property

Description Returns the name of the current Cal Type.

VB Syntax *value* = *meas*.**CalibrationName**

Variable [\(Type\)](#) - Description

value **(string)** - Variable to store the returned value.

meas A [Measurement](#) **(object)**

Return Type String

Default Not Applicable

Examples `ct = meas.CalibrationName`

C++ Syntax HRESULT get_CalibrationName(BSTR* CalibrationName);

Interface IMeasurement2

CalibrationPort Property - Obsolete

Description **Note:** Beginning with Rev 6.0, this command is no longer necessary. [Learn more](#). Because of improved calibration techniques, **Both** is always selected although a power meter measurement is performed only on port 1.

Specifies which SMC port to calibrate.

VB Syntax `SMC.CalibrationPort = value`

Variable [\(Type\)](#) - Description

SMC [SMCType](#) (object)

value (String) Port number to be calibrated. Choose from:

- 1 - SMC forward
- 2 - SMC reverse
- **Both**

Return Type String

Default 1

Examples `value = SMC.CalibrationPort = "Both"`

C++ Syntax `HRESULT put_CalibrationPort(BSTR port);`

`HRESULT get_CalibrationPort(BSTR *port);`

Interface `SMCType`

`VMCType`

CalibrationPorts Property

Description For each channel to be calibrated, sets and returns the ports to be calibrated. The port numbers need to be specified only for standard channels. Apps channels have designated input/output/LO ports.

Select any of the native PNA ports.

VB Syntax `calAll.CalibrationPorts (chan) = ports`

Variable [\(Type\)](#) - Description

calAll A [CalibrateAllChannels](#) (object)

chan Channel to be calibrated with a Calibrate All Channels calibration.

port (Variant Array) Ports to be calibrated for the specified channel.

Return Type Variant Array

Default Standard channels - ports 1 and 2

Apps channels - the designated input and output ports.

Examples

```
calAll.CalibrationPorts(2) = 0 'Ports to calibrate for channel 2
```

```
value = calAll.CalibrationPorts(2) 'returns the ports for channel 2
```

C++ Syntax HRESULT get_CalibrationPorts (long channel, Variant ports);

HRESULT put_CalibrationPorts (long channel, Variant* ports);

Interface ICalibrateAllChannels

Last modified:

4-Jan-2012 New topic

CalibrationTypeID Property

Description **Note:** This command replaces [Calibration Type Property](#).

Sets or returns the current cal type for the measurement using a Cal Type Name.

This command is used to set the Cal Type after recalling a Cal Set. [Learn more](#)

You can also use the CLSID or GUID associated with the Cal Type.

VB Syntax *meas*.**CalibrationTypeID** = *id*

Variable [\(Type\)](#) - Description

meas A [Measurement](#) (object)

id (**String**) Cal type. Case sensitive. Use one of the following:

For Full Calibrations:

This command does not distinguish between TRL and SOLT. The same number of error terms is applied for both Cal Types.

"Full n Port(x,y,z...)"

where

n = the number of ports to calibrate

x,y,z = the port numbers to calibrate

For example:

```
"Full 7 Port(2,3,4,5,6,7,8)"
```

For Response Calibrations:

"Response(param)" OR

"ResponseAndIsolation(param)"

Where param =

- S-parameter. For example"

- "Response (S21) "
- "ResponseAndIsolation(A/R) "
- Single or ratioed receivers using either [logical receiver notation](#) or physical receiver notation. For example:
 - "Response (A) "
 - "ResponseAndIsolation(a3/b4) "

For FCA Calibrations:

- "SMC_2P" (Response + Input + Output) All four sweeps required. Most accurate.
- "SMCRsp+IN" No Output match. All four sweeps required.
- "SMCRsp+OUT" No Output match. All four sweeps required.
- "SMCRsp" No Input or Output match. Saves two sweeps.

For VMC, multiple Cal types are not available.

For Gain Compression Cal

where r = receive port; s = source port

- "GCA 2P (r,s)" - full 2-port cal
- "GCA Enh Resp (r,s)" - Enhanced Response Cal

Return Type String

Default Not Applicable

Examples

```
Dim pna
Dim m

Set pna = CreateObject("AgilentPNA835x.Application")
Set m = pna.ActiveMeasurement
m.CalibrationTypeID = "Scalar Mixer Cal"
m.ErrorCorrection = True
MsgBox m.CalibrationName
```

C++ Syntax HRESULT get_CalibrationTypeID(BSTR* CalibrationTypeID);

HRESULT put_CalibrationTypeID(BSTR CalibrationTypeID);

Interface IMeasurement2

Last modified:

22-Sep-2009	Removed VMC strings
27-May-2008	Edit channel vs meas
11-Feb-2008	Fixed typo
9/12/06	MQ Added for multiport.

CalibrationFrequencies Property

Description Sets and returns the whether to perform the source power cal at the center frequencies midway between the main tones, or at all main tone frequencies.

VB Syntax *imd.CalibrationFrequencies = value*

Variable [\(Type\)](#) - Description

imd A [SweptIMDCal](#) (object)

value (Enum as NAIMDCalibrationFrequencies) Choose from:

0 - naIMDCenterFrequencies -Perform source power calibration at only the center frequencies midway between the main tones.

1 - naIMDALLFrequencies - Perform source power calibration at all main tone frequencies.

Return Type Enum

Default 0 - naIMDCenterFrequencies

Examples

```
imd.CalibrationFrequencies = naIMDALLFrequencies 'Write  
calFreq = imd.CalibrationFrequencies 'Read
```

C++ Syntax HRESULT get_CalibrationFrequencies(tagNAIMDCalibrationFrequencies * Val)
HRESULT put_CalibrationFrequencies(tagNAIMDCalibrationFrequencies newVal)

Interface ISweptIMD

Last Modified:

9-Sep-2008 MX New topic

CalKitType Property

Description Sets and returns a calibration kit type to be used for **UNGUIDED** calibration and for cal kit modification. To get a handle to this kit, use `app.ActiveCalKit` .

Although an unlimited number of cal kits can be imported into the PNA, **ONLY** mechanical cal kits #1 through #95 can be accessed.

There is also a CalKitType property for use during a Guided, SMC, and VMC Calibration.

VB Syntax `object .CalKitType = value`

Variable (Type) - Description

object calkit (object) or

Application (object)

Note: `app .CalKitType` and `calkit .calKitType` perform exactly the same function.

value (**enum naCalKit**) - Calibration Kit type. Choose from:

1 - naCalKit_User1
 2 - naCalKit_User2
 3 - naCalKit_User3
 4 - naCalKit_User4
 ..
 ..
 ..
 94 - naCalKit_User94
 95 - naCalKit_User95

These enumerated values correspond with the calibration kit ID on the Advanced Cal Kit Modify dialog box .

To change the cal kit name, use Name property .

Return Type NACalKit

Default Not Applicable

Examples `calkit.CalKitType = naCalKit_User27`

`kitype = app.CalKitType`

C++ Syntax HRESULT get_CalKitType(tagNACalKit *pVal);
HRESULT put_CalKitType(tagNACalKit newVal);

Interface IApplication
ICalKit

Last Modified:

12-Oct-2011 Increased limit to 95

CalKitType Property

Description Sets and returns the ECal or mechanical cal kit for the specified port number to be used during the calibration.

There is also a [CalKitType Property](#) for use during an Unguided Cal.

Note: Sliding loads are not fully supported from the GuidedCalibration object. The **Measure** button must be pressed manually on the PNA.

Note: This command replaces [Do1PortEcal Property](#) and [Do2PortEcal Property](#) for SMC and VMC Calibrations.

VB Syntax *object.CalKitType (port) = value*

Variable (Type) - Description

object Any of the following:

[GuidedCalibration](#) (object)

[SMCType](#) (object)

[VMCType](#) (object)

port (Long) Port number to which the cal kit will be assigned.

For Guided Cals and SMC, select port number.

For VMC calibrations:

- 1 - Mixer Input.
- Any unused port can be used for the mixer output.
- **Output port of MUT +1** - Output port of the calibration mixer. Generally this is port 3.

value (**string**) - Calibration Kit type. **Case-sensitive.**

Use [GetCompatibleCalKits](#) to return a list of valid Cal Kits.

Return Type String

Default Not Applicable

Examples

```
'Note: All of the following specify port 1 only
' Mechanical Cal Kit
```

```
guidedCal.CalKitType(1) = "85052C"
' Standard ECal modules
guidedCal.CalKitType(1) = "N4691-60004 ECal"
' Non-factory ECal characterizations are specified as follows:
guidedCal.CalKitType(1) = "N4691-60004 User 1 ECal"
' When two or more ECal modules with the same model number are
' connected, also specify the serial number as follows:
guidedCal.CalKitType(1) = "N4691-60004 ECal 01234"
' When Disk Memory ECal user characterizations are used,
' specify both the User char and the serial number as follows:
'guidedCal.CalKitType(1) = "N4691-60004 MyDskChar ECal 01234"
' Turn on auto orientation for the ECal (default behavior).
value = smc.CalKitType(1) 'Read
```

C++ Syntax HRESULT get_CalKitType(long port, BSTR *calkit)

HRESULT put_CalKitType(long port, BSTR calkit)

Interface IGuidedCalibration

SMCType

VMCType

Last Modified:

6-Apr-2011	Edited Ecal Examples
17-Feb-2011	Replaces Do1 and 2portEcal
13-Aug-2007	Added detail for port argument

CalKitType Property

Description	Sets and returns the name of the Cal Kit to use for unguided cal.
VB Syntax	<i>cal</i> .CalKitType (<i>port</i>) = <i>value</i>
Variable	(Type) - Description
<i>cal</i>	Calibrator (object)
<i>port</i>	(Long) Currently unused
<i>value</i>	(String) Cal Kit name enclosed in quotes. Use CalKitTypes to read a list of all available Cal Kits in the PNA.
Return Type	String
Default	Last kit selected
Examples	<pre>cal.CalKitType(1)= "85052B"</pre>
C++ Syntax	HRESULT put_CalKitType(long port, BSTR calKit); HRESULT get_CalKitType(long port, BSTR* pCalKit);
Interface	ICalibrator10

Last Modified:

17-Mar-2010 MX New topic

CalKitTypes Property

Description

VB Syntax *cal.CalKitTypes (port) = value*

Variable [\(Type\)](#) - Description

cal [Calibrator](#) (object)

port

value **(String)**

Return Type Long

Default

Examples `cal.CalKitTypes(4)=`

C++ Syntax HRESULT put_CalKitTypes(long port, BSTR calKit);

HRESULT get_CalKitTypes(long port, BSTR*
pCalKit);

Interface ICalibrator10

Last Modified:

17-Mar-2010 MX New topic

CalMethod Property

Description Sets and returns the method for performing calibration on a noise channel.

VB Syntax *noise*.**CalMethod** = *value*

Variable [\(Type\)](#) - Description

noise A [NoiseCal](#) (**object**)

value (**string**) Cal Method. Choose from:

- "VectorFull" or "Vector"
- "SParameter" (Not available for NFX measurements)
- "ScalarFull" or "Scalar"

Return Type String

Default "VectorFull"

Examples

```
noise.CalMethod = "VectorFull" 'Write
```

```
calMethod = noise.CalMethod 'Read
```

C++ Syntax HRESULT get_CalMethod(BSTR* pValue)

HRESULT put_CalMethod(BSTR pNewValue)

Interface INoiseCal

Last Modified:

27-Oct-2009 Updated for NFX

29-May-2007 MN New topic

CalMethod Property

Description Sets and returns the method by which the match-correction portion of an IMD calibration is performed. [Learn more.](#)

VB Syntax `imd.CalMethod = value`

Variable [\(Type\)](#) - Description

imd A [SweptIMDCal](#) (object)

value (Enum as NAIMDCalMethod) Choose from:

0 - naIMDMatchCorrectedResponse -Performs a full 2-port cal for full match-correction.

1 - naIMDResponseOnly - Performs only a response cal instead of a full 2-port cal.

Return Type Enum

Default 0 - **naIMDMatchCorrectedResponse**

Examples

```
imd.CalMethod = naIMDMatchCorrectedResponse 'Write
```

```
calMeth = imd.CalMethod 'Read
```

C++ Syntax HRESULT get_CalMethod(tagNAIMDCalMethod * Val)
HRESULT put_CalMethod(tagNAIMDCalMethod newVal)

Interface ISweptIMD

Last Modified:

18-Sep-2008 MX New topic

CalPower Property **Superseded**

Description This command is replaced by [SetCalInfoEx Method](#).

Specifies the power level that is expected at the desired reference plane (DUT input or output). This is not used for [segment sweep with independent power levels](#) or [power sweeps](#).

VB Syntax *value* = *powerCalibrator*.**CalPower** (*chan*, *sourcePort*)

Variable **(Type) - Description**

value **(double)** - Variable to store the returned Cal power value in dBm.

powerCalibrator **(object)** - A SourcePowerCalibrator object

chan **(long integer)** - Channel number of the PNA.

sourcePort **(long integer)** - Source port number.

Use [GetPortNumber](#) to return the port number of a source that only has a string name, such as an [External Source](#).

Return Type None

Default Not applicable

Examples

```
Set powerCalibrator = pna.SourcePowerCalibrator
power = powerCalibrator.CalPower(1,2) 'Read
```

C++ Syntax HRESULT get_CalPower(long channel, long sourcePort, double *pVal);

Interface ISourcePowerCalibrator

Last Modified:

7-Nov-2011 Fixed example

30-Apr-2007 Superseded

CalSet Property

Description	Set and read the Cal Set name into which the calibration will be saved. The phase reference cal can NOT be saved to a cal register. The Cal Set is saved by calling GenerateErrorTerms at the conclusion of the Guided Cal.
VB Syntax	<i>phasRef</i> .CalSet = <i>value</i>
Variable	(Type) - Description
<i>phasRef</i>	A PhaseReferenceCalibration Object
<i>value</i>	(String) Cal Set name.
Return Type	String
Default	Not Applicable
Examples	<pre>phase.CalSet = "PhaseRefCal"</pre> See example program
C++ Syntax	HRESULT get_CalSet(BSTR CalSet* pVals); HRESULT put_CalSet(BSTR CalSet newVals);
Interface	IPhaseReferenceCalibration

Last Modified:

3-Apr-2012 MX New topic

Center Property

Description Sets or returns the Center time of either Gating or Time Domain transform windows

VB Syntax *object.Center = value*

Variable [\(Type\) - Description](#)

object **(object)** As Gating

or

(object) As Transform

value **(double)** - Center time in seconds. Choose any number between:
 $\pm (\text{points}-1) / \text{frequency span}$

Return Type Double

Default 0

Examples

```
trans.Center = 4.5e-9 'sets the Center time of a transform window  
-Write  
gate.Center = 4.5e-9 'sets the Center time of a gating window -  
Write
```

```
cnt = trans.Center 'Read
```

C++ Syntax HRESULT get_Center(double *pVal)
HRESULT put_Center(double newVal)

Interface ITransform
IGating

Read-only

Center Property

Description Returns the stimulus value of the center data point for the measurement. This function does NOT work for segment sweep measurements. To understand how this property is useful, see [IMeasurement2 Interface](#).

VB Syntax *value* = *meas.Center*

Variable [\(Type\)](#) - Description

value **(Double)** - Variable to store the returned value.

meas A Measurement **(object)**

Return Type Double

Default Not Applicable

Examples `Print meas.Center 'prints the center data point`

C++ Syntax HRESULT get_Center(double * Val);

Interface IMeasurement2

CenterFrequency Property

Description Sets or returns the center frequency of the channel
or
Sets or returns the center frequency of the segment.

See the [Measurement2 Interface](#) to learn how this method differs from [meas.Center](#).

VB Syntax *object*.centerFrequency = *value*

Variable [\(Type\)](#) - Description

object A [Channel](#) (**object**)

or

A [Segment](#) (**object**)

value (**double**) - Center frequency in Hertz. Choose any number between the **minimum** and **maximum** frequencies of the analyzer.

Return Type Double

Default Center of the frequency range

Examples `chan.centerFrequency = 4.5e9 'sets the center frequency of a linear sweep for the channel object -Write`

`centfreq = chan.centerFrequency 'Read`

C++ Syntax HRESULT get_CenterFrequency(double *pVal)
HRESULT put_CenterFrequency(double newVal)

Interface IChannel
ISegment

ChannelNumber Property

Description Returns the Channel number of the Channel or Measurement object.

VB Syntax *object*.ChannelNumber

Variable [\(Type\)](#) - Description

object A Channel (**object**)
or
A Measurement (**object**)

Return Type Long Integer

Default Not applicable

Examples

```
chanNum = chan.ChannelNumber 'returns the channel number
chanNum = meas.ChannelNumber 'returns the channel number of the
measurement
```

C++ Syntax HRESULT get_ChannelNumber(long *pVal)

Interface IChannel
IMeasurement

Channels Property

Description	Sets and returns the list of channels to be calibrated during the Cal All session.
VB Syntax	<i>calAll.Channels</i> = <i>chans</i>
Variable	(Type) - Description
<i>calAll</i>	A CalibrateAllChannels (object)
<i>chans</i>	(Variant) Array of channel numbers to be calibrated, separated by commas. These channels must already exist.
Return Type	Variant
Default	The existing channels
Examples	<pre>calAll.Channels = Array(1,2,3) 'sets channels to be cal'd</pre> <pre>chans = calAll.Channels 'returns the channel numbers to be cal'd</pre>
C++ Syntax	HRESULT get_Channels (VARIANT selectedChannels); HRESULT put_Channels (VARIANT* selectedChannels);
Interface	ICalibrateAllChannels

Last modified:

4-Jan-2012 New topic

CharacterizationNumber Property

Description Sets and reads the number to which the user characterization will be stored in the ECal module. The number must be set before sending [Initialize](#) or the default value (1) will be used.

VB Syntax *userChar.CharacterizationNumber = value*

Variable [\(Type\)](#) - Description

userChar An [ECalUserCharacterizer](#) Object

value (Long) User Characterization number. Choose a value between 1 and 12.

Return Type Long Integer

Default Not Applicable

Examples `userChar.CharacterizationNumber = 5`

C++ Syntax HRESULT put_CharacterizationNumber(long *Number);

Interface IECalUserCharacterizer

Last Modified:

2-Nov-2008 New topic (8.33)

CharacterizeMixerOnly Property

Description Sets and returns whether to perform ONLY a mixer characterization.

VB Syntax `VMC.CharacterizeMixerOnly = bool`

Variable [\(Type\)](#) - Description

`VMC` [VMCType](#) (object)

`bool` (Boolean)

True - Perform ONLY mixer characterization.

False - Perform both mixer characterization and calibration.

Return Type Boolean

Default **False**

Examples `value = VMC.CharacterizeMixerOnly`

C++ Syntax `HRESULT put_CharacterizeMixerOnly(VARIANT_BOOL
bCharMixerOnly);`

`HRESULT get_CharacterizeMixerOnly(VARIANT_BOOL
*bCharMixerOnly);`

Interface VMCType

CharFileName Property

Description	Specifies the mixer characterization (.S2P) file and immediately loads the file. Also specify the use of a characterization file with LoadCharFromFile Property
VB Syntax	<i>VMC.CharFileName</i> = <i>value</i>
Variable	(Type) - Description
<i>VMC</i>	VMCType (object)
<i>value</i>	(String) Full path, file name, and extension of the mixer characterization file.
Return Type	Not Applicable
Default	Not Applicable
Examples	<pre>VMC.CharFileName = "c:\users\public\network analyzer\documents/default.S2P"</pre>
C++ Syntax	<pre>HRESULT put_CharFileName(BSTR filename); HRESULT get_CharFileName(BSTR *filename);</pre>
Interface	VMCType

CharMixerReverse Property

Description	Specifies the direction in which to characterize the calibration mixer. Learn more about the calibration mixer.
VB Syntax	<code>VMC.CharMixerReverse = bool</code>
Variable	(Type) - Description
<code>VMC</code>	VMCType (object)
<code>bool</code>	(Boolean)
	0 - Characterize the calibration mixer in the SAME direction as that specified in the mixer setup.
	1 - Characterize the calibration mixer in the REVERSE direction as that specified in the mixer setup.
Return Type	Boolean
Default	0
Examples	<pre>VMC.CharMixerReverse = 0</pre>
C++ Syntax	<pre>HRESULT put_CharMixerReverse(VARIANT_BOOL bcharReverse); HRESULT get_CharMixerReverse(VARIANT_BOOL *bcharReverse);</pre>
Interface	VMCType2

CitiContents Property - Superseded

Description	This command is replaced with SaveData Method Specifies the contents of subsequent citifile saves using app. SaveCitiDataData or app. SaveCitiFormattedData
VB Syntax	<i>pref.CitiContents</i> = <i>value</i>
Variable	(Type) - Description
<i>pref</i>	A Preferences (object)
<i>value</i>	(string) - Contents that will be saved with subsequent save commands. Choose from: "Single" - Single trace "Displayed" - All displayed traces "Auto" - All displayed traces
Return Type	String
Default	"Auto"
Examples	<pre>pref.CitiContents = "Single" 'Write content = pref.CitiContents 'Read</pre>
C++ Syntax	HRESULT get_CitiContents(BSTR *Contents) HRESULT put_CitiContents(BSTR Contents)
Interface	IPreferences

CitiFormat Property - Superseded

Description	This command is replaced with SaveData Method Specifies the format of subsequent citifile saves using app. SaveCitiFormattedData
VB Syntax	<i>pref.CitiFormat</i> = <i>value</i>
Variable	(Type) - Description
<i>pref</i>	A Preferences (object)
<i>value</i>	(string) - Format in which the citifile will be saved with subsequent save commands. Choose from: "MA" - Linear Magnitude / degrees "DB" - Log Mag / degrees "RI" - Real / Imaginary "Auto" - Format in which the trace is already displayed. If other than Log Mag, Linear Magnitude, or Real/Imag, then the format will be in Real/Imag.
Return Type	String
Default	"Auto"
Examples	<pre>pref.CitiFormat = "MA" 'Write format = pref.CitiFormat 'Read</pre>
C++ Syntax	HRESULT get_CitiFormat(BSTR *Format) HRESULT put_CitiFormat(BSTR Format)
Interface	IPreferences

CmnModeZConvPortImag Property

Description Sets the imaginary part of the impedance value for the common port impedance conversion function.

VB Syntax `fixture.CmnModeZConvPortImag(portNum) = value`

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

portNum **(Integer)** Balanced (logical) port number. Choose from logical ports 1, 2, or 3. [Learn more about logical ports.](#)

value **(Double)** Imaginary part of the Impedance value. Choose a value between 0 and 1E18.

Return Type Double

Default 0

Examples `fixture.CmnModeZConvPortImag(2) = 75 'Write`

`value = fixture.CmnModeZConvPortImag(1) 'Read`

C++ Syntax HRESULT get_CmnModeZConvPortImag(short portNum, double *pVal)
HRESULT put_CmnModeZConvPortImag(short portNum, double newVal)

Interface IFixturing2

CmnModeZConvPortReal Property

Description Sets the real part of the impedance value for the common port impedance conversion function.

VB Syntax `fixture.CmnModeZConvPortReal(portNum) = value`

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

portNum (**Integer**) Balanced (logical) port number. Choose from logical ports 1, 2, or 3. [Learn more about logical ports.](#)

value (**Double**) Real part of the Impedance value. Choose a value between 0 and 1E18.

Return Type Double

Default See [Common Mode Port Z Conversion Default](#)

Examples

```
fixture.CmnModeZConvPortReal(2) = 75 'Write
```

```
value = fixture.CmnModeZConvPortReal(1) 'Read
```

C++ Syntax HRESULT get_CmnModeZConvPortReal(short portNum, double *pVal)

HRESULT put_CmnModeZConvPortReal(short portNum, double newVal)

Interface IFixturing2

Last Modified:

31-May-2013 Fixed Typo

CmnModeZConvPortZ0 Property

Description Sets the impedance value for the common port impedance conversion function. Set either this single value or set the [real](#) and [imaginary](#) parts separately. The imaginary part is set to 0.0 using this command.

VB Syntax `fixture.CmnModeZConvPortZ0(portNum) = value`

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

portNum (**Integer**) Balanced (logical) port number. Choose from logical ports 1, 2, or 3. [Learn more about logical ports.](#)

value (**Double**) Impedance value. Choose a value between 0 and 1E7.

Return Type
Double

Default See [Common Mode Port Z Conversion Default](#)

Examples

```
fixture.CmnModeZConvPortZ0(2) = 75 'Write  
value = fixture.CmnModeZConvPortZ0(1) 'Read
```

C++ Syntax HRESULT get_CmnModeZConvPortZ0(short portNum, double *pVal)

HRESULT put_CmnModeZConvPortZ0(short portNum, double newVal)

Interface IFixturing2

Last Modified:

31-May-2013 Fixed Typo

CmnModeZConvState Property

Description Turns ON or OFF 4-port common port impedance conversion function. Must also set the fixture simulator function to ON using [FixturingState Property](#).

VB Syntax *fixture.CmnModeZConvState = value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

value (**Boolean**)

False - Turns common port impedance conversion OFF

True - Turns common port impedance conversion ON

Return Type Boolean

Default False

Examples

```
fixture.CmnModeZConvState = False 'Write
```

```
value = fixture.CmnModeZConvState 'Read
```

C++ Syntax HRESULT get_CmnModeZConvState(VARIANT_BOOL *pVal)

HRESULT put_CmnModeZConvState(VARIANT_BOOL newVal)

Interface IFixturing2

CompatibleCalKits Property

Description Returns a comma-separated list of valid kits that use the specified connector type. This includes mechanical cal kits, applicable characterizations found within ECal modules currently connected to the PNA, and all user characterizations stored in PNA disk memory.

Note: Beginning with PNA Rev 9.1, the serial number is returned for ALL ECal modules that are connected with the connector type of the specified port. Previously, the returned list would include the serial numbers to distinguish the ECal modules only when two or more identical ECal models were connected to the PNA.

VB Syntax *value* = *obj*.CompatibleCalKits (*port*)

Variable [\(Type\)](#) - Description

value (Variant) Variable to store the returned list of Cal Kits.

obj Any of the following:

[GuidedCalibration](#) (object) **Superseded** Replaced with [GetCompatibleCalKits Method](#)

[SMCType](#) (object)

[VMCType](#) (object)

port (Long) Port number for which you want compatible kits.

First set the [ConnectorType](#) for the port.

Return Type Variant

Default Not Applicable

Examples

```
Dim kits As Variant
kits = MySMC.CompatibleCalKits(1)
```

C++ Syntax HRESULT get_CompatibleCalKits(long port, VARIANT* Kits);

Interface IGuidedCalibration

SMCType

VMCType

16-Nov-2009 Added Note (9.1)

16-Nov-2009 Superseded

CompositeNormalizationMode Property

Description Sets and returns the method by which CTB and CSO calculations are performed.

VB Syntax *imd.CompositeNormalizationMode* = *value*

Variable [\(Type\)](#) - Description

imd A [SweptIMD](#) Object

value (Enum)

0 - naNone - the normalized power is not used in calculation

1 - naNumberOfCarriers - CTB and CSO is corrected by subtracting $10 \cdot \log(N/2)$, where

- N = # of carriers for CTB
- N = # of distortion products for CSO

2 - naPdBm - the composited normalized power for CTB or CSO is treated as a dBm value

3 - naPdBmV - the composited normalized power for CTB or CSO is treated as a dBmV value.

Note: Power values are stored using the currently-set units. Therefore, first set units with this command, then set power values using:

[CompositeNormalizedCSOPower](#)

[CompositeNormalizedCTBPower](#)

Return Type Enum

Default **naNumberOfCarriers**

Examples `imd.CompositeNormalizationMode = naNone 'Write`

`value = imd.CompositeNormalizationMode 'Read`

C++ Syntax HRESULT

get_CompositeNormalizationMode(tagNAIMDCompositeNormalizationMode *pVal)

HRESULT

put_CompositeNormalizationMode(tagNAIMDCompositeNormalizationMode pVal)

Interface ISweptIMD

Last Modified:

16-Sep-2008 MX New topic

CompositeNormalizedCSOPower Property

Description Sets and returns the CSO Power for POWER [normalization mode](#). Valid only with measurement parameters: CSO2Lo and CSO2Hi and for Normalization Modes **dBm** and **dBmV**.

VB Syntax *imd.CompositeNormalizedCSOPower = value*

Variable [\(Type\)](#) - Description

imd A [SweptIMD](#) Object

value (Double) Power level. The units are determined by [CompositeNormalizationMode Property](#), which must be set first.

Return Type Double

Default 0

Examples

```
imd.CompositeNormalizedCSOPower = -5 'Write
```

```
value = imd.CompositeNormalizedCSOPower 'Read
```

C++ Syntax HRESULT get_CompositeNormalizedCSOPower(double *pVal)
HRESULT put_CompositeNormalizedCSOPower(double pVal)

Interface ISweptIMD

Last Modified:

16-Sep-2008 MX New topic

CompositeNormalizedCTBPower Property

Description Sets and returns the CSO Power. Valid only with measurement parameters: CTBLo and CTBHi and for Normalization Modes **dBm** and **dBmV**.

VB Syntax *imd.CompositeNormalizedCTBPower = value*

Variable [\(Type\)](#) - Description

imd A [SweptIMD](#) Object

value (Double) Power level. The units are determined by [CompositeNormalizationMode Property](#), which must be set first.

Return Type Double

Default 0

Examples

```
imd.CompositeNormalizedCTBPower = -5 'Write
```

```
value = imd.CompositeNormalizedCTBPower 'Read
```

C++ Syntax HRESULT get_CompositeNormalizedCTBPower(double *pVal)
HRESULT put_CompositeNormalizedCTBPower(double pVal)

Interface ISweptIMD

Last Modified:

7-Apr-2009 Corrected parameters

16-Sep-2008 MX New topic

Compression Property

Description	Returns the Compression result of the PNOP marker search. Pnop Comp = Pnop Gain - Linear Gain (not shown on marker readout).
VB Syntax	<i>comp</i> = <i>pnop</i> . Compression
Variable	(Type) - Description
<i>comp</i>	(double) - Variable to store returned value
<i>pnop</i>	A PNOP (object)
Return Type	Double
Default	Not applicable
Examples	<pre>comp = pnop.Compression 'Read</pre> See example program
C++ Syntax	HRESULT get_Compression(double* pNewVal)
Interface	IPNOP

Last Modified:

19-Feb-2010 MX New topic

CompressionAlgorithm Property

Description Set and read the algorithm method used to compute gain compression.

VB Syntax `gca.CompressionAlgorithm = value`

Variable [\(Type\)](#) - Description

gca A [GainCompression](#) (object)

value (**tagNAGCACCompressionAlgorithm**) - Algorithm method. Choose from:

- **naCompressionFromLinearGain (0)**
- **naCompressionFromMaximumGain (1)**
- **naBackoffCompression (2)**
- **naXYCompression (3)**
- **naSaturation (4)**

Return Type Enum

Default `naCompressionFromLinearGain (0)`

Examples

```
gca.CompressionAlgorithm = naXYCompression 'Write  
compAlg = gca.CompressionAlgorithm 'Read
```

C++ Syntax `HRESULT get_CompressionAlgorithm(tagNAGCACCompressionAlgorithm* pVal)`
`HRESULT put_CompressionAlgorithm(tagNAGCACCompressionAlgorithm newVal)`

Interface `IGainCompression`

Last Modified:

11-Sep-2007 MX New topic

CompressionBackoff Property

Description Set and read value for the BackOff compression algorithm.

VB Syntax `gca.CompressionBackoff = value`

Variable [\(Type\)](#) - Description

gca A [GainCompression](#) (object)

value (**Double**) Backoff value in dB. Choose from 30 to (-30)

Return Type Double

Default 10

Examples `gca.CompressionBackoff = 7 'Write`

`acqMode = gca.CompressionBackoff 'Read`

C++ Syntax HRESULT get_CompressionBackoff(double* pValue)

HRESULT put_CompressionBackoff(double newValue)

Interface IGainCompression

Last Modified:

21-Nov-2007 MX New topic

CompressionDeltaX Property

Description Set and read the 'X' value in the delta X/Y compression algorithm.

VB Syntax `gca.CompressionDeltaX = value`

Variable [\(Type\)](#) - Description

gca A [GainCompression](#) (object)

value (**double**) X value in dB. Choose from 30 to (-30)

Return Type Double

Default 10

Examples

```
gca.CompressionDeltaX = 'Write
```

```
xDelta = gca.CompressionDeltaX 'Read
```

C++ Syntax HRESULT get_CompressionDeltaX(double* pVal)

HRESULT put_CompressionDeltaX(double newVal)

Interface IGainCompression

Last Modified:

11-Sep-2007 MX New topic

CompressionDeltaY

Description Set and read the 'Y' value in the delta X/Y compression algorithm.

VB Syntax `gca.CompressionDeltaY = value`

Variable [\(Type\)](#) - Description

gca A [GainCompression](#) (object)

value (double) -

Return Type Double

Default 9

Examples `gca.CompressionDeltaY = 7 'Write`

`xDelta = gca.CompressionDeltaY 'Read`

C++ Syntax HRESULT get_CompressionDeltaY(double* pVal)

HRESULT put_CompressionDeltaY(double newVal)

Interface IGainCompression

Last Modified:

11-Sep-2007 MX New topic

CompressionInterpolation Property

Description Sets whether or not interpolation should be performed on 2D measured compression data. Applies ONLY to [2D acquisition modes](#).

VB Syntax `gca.CompressionInterpolation = value`

Variable [\(Type\)](#) - Description

gca A [GainCompression](#) (object)

value (boolean) - Choose from:

True Interpolate the results

False Do NOT interpolate the results but return the value closest to compression.

Return Type Boolean

Default False

Examples

```
gca.CompressionInterpolation = True 'Write
compInt = gca.CompressionInterpolation 'Read
```

C++ Syntax HRESULT get_CompressionInterpolation(VARIANT_BOOL* pVal)
HRESULT put_CompressionInterpolation(VARIANT_BOOL newVal)

Interface IGainCompression

Last Modified:

11-Sep-2007 MX New topic

CompressionLevel Property

Description Set and read the desired gain reduction (from reference gain).

This value is used for [Compression Methods](#): Compression from Linear Gain and Compression from Maximum Gain.

VB Syntax `gca.CompressionLevel = value`

Variable (Type) - Description

gca A [GainCompression](#) (object)

value (**Double**) - Compression level in dB. Choose a value greater than 0.1 dB.

Return Type Double

Default 1

Examples `gca.CompressionLevel = 1.5 'Write`

`compLevel = gca.CompressionLevel 'Read`

C++ Syntax HRESULT get_CompressionLevel(double* pVal)

HRESULT put_CompressionLevel(double newVal)

Interface IGainCompression

Last Modified:

11-Sep-2007 MX New topic

CompressionMax Property

Description Returns the Compression Max result of a PSat or PNOP marker search.

Comp Max = Gain Max - Linear Gain (not shown on PNOP marker readout).

VB Syntax *compMax* = *pMarker*.**CompressionMax**

Variable [\(Type\)](#) - Description

compMax (**double**) - Variable to store returned value

pMarker A [PNOP](#) (object) or a [PSaturation](#) (object)

Return Type Double

Default Not applicable

Examples `compMax = pMark.CompressionMax 'Read`

[See example program](#)

C++ Syntax HRESULT get_CompressionMax(double* pNewVal)

Interface IPNOP or IPSaturation

Last Modified:

19-Feb-2010 MX New topic

CompressionLevel (Marker) Property

Description	Set and read the marker compression level. First use SearchCompressionPoint to create the compression marker.
VB Syntax	<i>mkr.CompressionLevel = value</i>
Variable	(Type) - Description
<i>mkr</i>	A Marker (object)
<i>value</i>	(Double) - Compression level in dB. Choose any number between: -500 dB to 500 dB Standard gain compression values are positive.
Return Type	Double
Default	1
Examples	See example program
C++ Syntax	HRESULT get_CompressionLevel(double* pVal) HRESULT put_CompressionLevel(double newVal)
Interface	IMarker4

Last Modified:

8-Feb-2009 MX New topic

CompressionPin Property

Description	Reads the input power at the marker compression level. First issue SearchCompressionPoint Method or Tracking Property .
VB Syntax	<i>value</i> = <i>mkr</i> . CompressionPin
Variable	(Type) - Description
<i>mkr</i>	A Marker (object)
<i>value</i>	(Double) - Variable to store the returned input power value.
Return Type	Double
Default	Not Applicable
Examples	<pre>compLevel=mkr.CompressionPin 'Read</pre> See example program
C++ Syntax	HRESULT get_CompressionPin(double* pVal)
Interface	IMarker4

Last Modified:

8-Feb-2009 MX New topic

CompressionPout Property

Description	Reads the output power at the marker compression level. First issue SearchCompressionPoint Method or Tracking Property .
VB Syntax	<i>value</i> = <i>mkr</i> . CompressionPout
Variable	(Type) - Description
<i>mkr</i>	A Marker (object)
<i>value</i>	(Double) - Variable to store the returned output power value.
Return Type	Double
Default	Not Applicable
Examples	<pre>compLevel=mkr.CompressionPout 'Read</pre> See example program
C++ Syntax	HRESULT get_CompressionPout(double* pVal)
Interface	IMarker4

Last Modified:

8-Feb-2009 MX New topic

CompressionSaturation Property

Description Returns the Compression Saturation result of a PSat marker search.

Comp Sat = Gain Sat - Gain Linear

VB Syntax *compSat* = *pSat*.**CompressionSaturation**

Variable [\(Type\)](#) - Description

compSat (**double**) - Variable to store returned value

pSat A [PSaturation](#) (object)

Return Type Double

Default Not applicable

Examples `compSat = pSat.CompressionSaturation 'Read`

[See example program](#)

C++ Syntax HRESULT get_CompressionSaturation(double* pNewVal)

Interface IPSaturation

Last Modified:

22-Feb-2010 MX New topic

ConnectorType Property

Description Sets or queries the connector type for the specified port.

VB Syntax `obj.ConnectorType (port) = value`

Variable [\(Type\) - Description](#)

obj Any of the following:

[GuidedCalibration](#) (object)

[SMCType](#) (object)

[VMCType](#) (object)

port (Long) Port number of the connector type.

For Guided Cals and SMC, select port number.

For VMC calibrations:

- 1 - Mixer Input.
- Any unused port can be used for the mixer output.
- Output port of MUT +1 - Output port of the calibration mixer. Generally this is port 3.

value (String) - Connector type. **Case-sensitive.**

Use [ValidConnectorType Property](#) to list connector types.

Return Type String

Default None

Examples

```
SMC.ConnectorType(1) = "APC 3.5 male"
value = SMC.ConnectorType(1)
```

C++ Syntax HRESULT get_ConnectorType(long port, BSTR *connector)

HRESULT put_ConnectorType(long port, BSTR connector)

Interface IGuidedCalibration

SMCType

VMCType

Last Modified:

1-Dec-2011	Updated with VMC
13-Aug-2007	Added detail to port argument

ConnectorType Property

Description Sets or queries the connector type for the specified port.

VB Syntax *ecalUser.ConnectorType (port) = value*

Variable [\(Type\) - Description](#)

ecalUser An [ECalUserCharacterizer](#) (object)

port (Enum) ECal port for which connector type is to be set. Choose from:

1 or **naECalPort_A**

2 or **naECalPort_B**

3 or **naECalPort_C**

4 or **naECalPort_D**

value (String) - Connector type.

When the User Characterization is to be stored in the ECal module, then the connector type is limited to a Factory-defined connector type. [See the list](#).

When the User Characterization is to be stored in PNA disk memory, then the connector type can also be a User-defined connector type.

Return Type String

Default "" (Empty String)

Examples

```
ecalUser.ConnectorType(naECalPort_B) = "APC 3.5 male" ' Write
```

```
Value = ecalUser.ConnectorType(naECalPort_B)
```

C++ Syntax

```
HRESULT get_ConnectorType(NAECalPort port, BSTR *connector);
```

```
HRESULT put_ConnectorType(NAECalPort port, BSTR connector);
```

Interface IECalUserCharacterizer

Last Modified:

18-Sep-2009 Modified for factory connectors (A.09.00)

5-Nov-2008 New topic (8.33)

ControlLines Property

Description Sets the control lines of the specified test set. Control lines, provided through the front panel connector of a test set, are used to control external equipment such as a part handler. See your test set documentation to learn more about control lines.

VB Syntax *tset*.ControlLines (*chNum*) = *value*

Variable [\(Type\)](#) - Description

tset A [TestsetControl](#) object.

OR

An [E5091Testset](#) object.

chNum **(Integer)** Channel number of the measurement.

value **(Double)** Data value used to set control lines. Values are obtained by adding weights from the following table that correspond to individual lines.

Line	Weight
1	1
2	2
3	4
4	8
5	16
6	32
7	64
8	128

- The E5091A interprets SENS:MULT1:OUTP 0 as all lines LOW.
- All "Z" and "H" series test sets interpret SENS:MULT1:OUTP 0 as all lines HIGH.

Refer to your test set documentation for setting control line values.

Return Type Variant

Default 0

Examples `'For a Z5623A K64 test set, the following sets line 3 and 4 OFF; all other lines ON.'`

```
testset1.ControlLines(2) = 12
```

[See E5091A Example Program](#)

[See External Testset Program](#)

C++ Syntax HRESULT get_ControlLines(long channelNum, VARIANT *stateByte);
HRESULT put_ControlLines(long channelNum, VARIANT stateByte);

Interface ITestsetControl

IE5091Testset

Last Modified:

17-Aug-2007 Added different test sets active High and low

Read-only

Count Property

Description Returns the number of items in a collection of objects.

VB Syntax *object*.Count

Variable [\(Type\)](#) - Description

object Any of the following **(objects)**:

- [CalFactorSegments Collection](#)
- [CalFactorSegmentsPMAR Collection](#)
- [Cal Sets Collection](#)
- [Channels Collection](#)
- [E5091Testset Collection](#)
- [ExternalDevices Collection](#)
- [ExternalTestsets Collection](#)
- [GuidedCalibrationPowerSensors Collection](#)
- [LimitTest Collection](#)
- [Measurements Collection](#)
- [NaWindows Collection](#)
- [PowerLossSegments Collection](#)
- [PowerLossSegmentsPMAR_Collection.htm](#)
- [PowerSensors Collection](#)
- [Segments Collection](#)
- [Traces Collection](#)
- [PowerMeterInterfaces Collection](#)

Return Type Long Integer

Default Not applicable

Examples

```
numofchans = chans.Count 'return the number of channels  
-Read
```

[C++ Syntax](#) HRESULT get_Count(long *pInterface)

Interface All listed above

Last Modified:

8-Feb-2011	Added GuidedCalibratioPowerSensors (9.33)
31-Jul-2009	Added ExternalDevices (9.0)
6-Mar-2009	PowerMeterInterfaces

CouplePorts Property

Description Turns ON and OFF port power coupling. ON means the power level is the same for both ports. OFF means the power level may be set independently for each port.

VB Syntax *object.CouplePorts* = *value*

Variable [\(Type\)](#) - Description

object Channel (**object**)

or

CalSet (**object**) - Read-only property

value (**enum NAStates**) Choose from:

0 - **NaOff** - Turns coupling OFF

1 - **NaOn** - Turns coupling ON

Return Long Integer

Type **1** - ON

0 - OFF

Default NaON (1)

Examples

```
chan.CouplePorts = NaOff 'Write
```

```
couplport = chan.CouplePorts 'Read
```

C++ Syntax HRESULT get_CouplePorts(tagNAStates *pState)
 HRESULT put_CouplePorts(tagNAStates newState)

Interface IChannel

|CalSet3

CoupleChannelParams Property

Description Turns ON and OFF Time Domain Trace Coupling. All of the measurements in the specified channel are coupled.

- To select Transform parameters to couple, use [Trans.CoupledParameters Property](#)
- To select Gating parameters to couple, use [Gate.CoupledParameters Property](#)

VB Syntax *chan.CoupleChannelParams = state*

Variable [\(Type\)](#) - Description

chan A [Channel](#) (object)

state (**boolean**)
False - Turns Trace Coupling OFF

True - Turns Trace Coupling ON

Return Type Boolean

Default True

Examples `chan.CoupleChannelParams = False 'Write`

`couple = chan.CoupleChannelParams 'Read`

C++ Syntax HRESULT get_CoupleChannelParams(VARIANT_BOOL *isCoupled);

HRESULT put_CoupleChannelParams(VARIANT_BOOL isCoupled);

Interface IChannel5

Coupled Property

Description	Sets and returns the state of coupling (ON or OFF) of this range to the primary range.
VB Syntax	<i>FOMRange.Coupled</i> = <i>value</i>
Variable	(Type) - Description
<i>object</i>	An FOMRange (object)
<i>value</i>	(boolean) - State of coupling. True - Couple range to primary range. False - Do NOT couple to primary range.
Return Type	Boolean
Default	True
Examples	<pre>fomRange.Coupled = False 'this range is NOT coupled to the primary range.</pre> <pre>coupl = fomRange.Coupled 'Read</pre>
C++ Syntax	HRESULT get_Coupled(VARIANT_BOOL *pVal) HRESULT put_Coupled(VARIANT_BOOL pVal)
Interface	IFOMRange

CoupledMarkers Property

Description Sets and Reads the state of Coupled Markers (ON and OFF)

VB Syntax `app.CoupledMarkers = state`

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

state (boolean)
False (0) - Turns Coupled Markers OFF
True (1) - Turns Coupled Markers ON

Return Type Boolean
False - OFF
True - ON

Default **False**

Examples
`app.CoupledMarkers = True 'Write`
`coupl = app.CoupledMarkers 'Read`

C++ Syntax HRESULT put_CoupledMarkers(VARIANT_BOOL bState)
HRESULT get_CoupledMarkers(VARIANT_BOOL *bState)

Interface IApplication

CoupledParameters Property (Gating)

Description Specifies the time domain gating parameters to be coupled. The settings for those parameters will be copied from the active measurement to all other measurements on the channel.

To turn coupling ON and OFF, use [CoupleChannelParams Property](#)

To specify Transform parameters to couple, use [Transform.CoupledParameters Property](#)

VB Syntax `gate.CoupledParameters = value`

Variable [\(Type\)](#) - Description

trans A [Gating](#) (object)

value (**Enum As NAGatingCoupledParams**) - Parameters to couple. To specify more than one parameter, add the numbers. Choose from:

1 - naGatingStimulusCoupled (Start, Stop, Center, and Span TIME settings.)

2 - naGateStateCoupled (ON / OFF)

4 - naGatingShapeCoupled (Minimum, Normal, Wide, and Maximum)

8 - naGatingTypeCoupled (Bandpass and Notch)

Return Type Enum

Default 29

Examples `gate.CoupledParameters = 15 'Couple all parameters`

`CP = gate.CoupledParameters 'Read`

C++ Syntax `HRESULT get_CoupledParameters(long *IParams);`

`HRESULT put_CoupledParameters(long IParams);`

Interface IGating2

CoupledParameters Property (Transform)

Description Specifies the time domain transform parameters to be coupled. The settings for those parameters will be copied from the active measurement to all other measurements on the channel.

To turn coupling ON and OFF, use [CoupleChannelParams Property](#)

To specify Gating parameters to couple, use [Gate.CoupledParameters Property](#)

VB Syntax *trans.CoupledParameters* = *value*

Variable [\(Type\)](#) - Description

trans A [Transform](#) (object)

value **(Enum As NATransformCoupledParams)** - Parameters to couple. To specify more than one parameter, add the numbers. Choose from:

1 - naTransformStimulusCoupled (Start, Stop, Center, and Span TIME settings.)

2 - naTransformStateCoupled (ON / OFF)

4 - naTransformWindowCoupled (Kaiser Beta / Impulse Width)

8 - naTransformModeCoupled (Low Pass Impulse, Low Pass Step, Band Pass)

16 - naTransformDistMkrUnitCoupled (Distance maker Units)

Return Type Enum

Default 29

Examples `trans.CoupledParameters = 31 'Couple all parameters`

`CP = trans.CoupledParameters 'Read`

C++ Syntax HRESULT get_CoupledParameters(long *IParams);

HRESULT put_CoupledParameters(long IParams);

Interface ITransform2

CouplePhasePortSettings Property

Description	Sets and returns whether to couple phase control settings (IFBW, Tolerance, Max Iterations).
VB Syntax	<i>phase.CouplePhasePortSettings(srcPort) = value</i>
Variable	(Type) - Description
<i>phase</i>	A PhaseControl Object
<i>srcPort</i>	(Long Integer) Source port for which to make phase control settings.
	<p>Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use chan.getPortNumber to translate the string into a port number. To learn more see Remotely Specifying a Source Port</p>
<i>value</i>	(Boolean) Coupling state. Choose from: <p>True - Couple phase control settings. The phase control settings from <port> are copied to the other phase-controlled ports.</p> <p>False - Do NOT couple phase control settings. The phase control settings for each phase-controlled port are made independently.</p>
Return Type	Boolean
Default	False
Examples	<pre>phase.CouplePhasePortSettings 1 = True ' Write value = phase.CouplePhasePortSettings 2 ' Read</pre>
C++ Syntax	<pre>HRESULT get_CouplePhasePortSettings(long port, VARIANT_BOOL* pVal); HRESULT put_CouplePhasePortSettings(long port, VARIANT_BOOL newVal);</pre>
Interface	IPhaseControl

Last Modified:

8-Dec-2010 MX New topic

CoupleTonePower Property

Description Sets and returns the ON | OFF state of power coupling for F1 and F2.

VB Syntax *object.CoupleTonePower* = *value*

Variable [\(Type\)](#) - Description

object A [SweptIMD](#) or [IIMSpectrum](#) Object

value (Boolean) - Choose from:

True - F1 and F2 power is coupled.

False - F1 and F2 power is NOT coupled. Set power levels individually.

Return Type Boolean

Default True

Examples `ims.CoupleTonePower = true 'Write`

`value = ims.CoupleTonePower 'Read`

C++ Syntax HRESULT get_CoupleTonePower(VARIANT_BOOL* val)
HRESULT put_CoupleTonePower(VARIANT_BOOL val)

Interface ISweptIMD

IIMSpectrum

Last Modified:

19-Aug-2008 MX New topic

CpuRevision Property

Description	Returns a number that corresponds to the CPU speed of the PNA.
VB Syntax	<i>value</i> = <i>cap</i> . CpuRevision
Variable	(Type) - Description
<i>value</i>	(String) - Variable to store the returned number. Use the following table to learn the CPU speed. Reported CPU version - Clock speed 1.0 - 266 MHz 2.0 - 500 MHz 3.0 - 1100 MHz 4.0 - 1600 MHz 5.0 - 2000 MHz 6.0 - 2000 MHz dual core.
<i>cap</i>	A Capabilities (object)
Return Type	String
Default	Not Applicable
Examples	<code>value = cap.CpuRevision 'Read</code>
C++ Syntax	HRESULT get_CpuRevision(BSTR *value);
Interface	ICapabilities6

Last Modified:

5-Jan-2012	Added 6.0 dual core
30-Nov-2009	MX New topic

CSONumDistortionProducts Property

Description Sets and returns the “N = number of distortion products” value for the calculation of the CSO parameter.
[Learn more.](#)

VB Syntax *imd.CSONumDistortionProducts = value*

Variable [\(Type\)](#) - Description

imd A [SweptIMD](#) Object

value (Long Integer) Number of distortion products

Return Type Long Integer

Default 40

Examples

```
imd.CSONumDistortionProducts = True 'Write
```

```
value = imd.CSONumDistortionProducts 'Read
```

C++ Syntax HRESULT get_CSONumDistortionProducts(long *pVal)
HRESULT put_CSONumDistortionProducts(long *pVal)

Interface ISweptIMD

Last Modified:

9-Jun-2010 Fixed type
25-Aug-2008 MX New topic

CSOOffset Property

Description Sets and returns the offset that is applied to CSO measurements. Valid only with measurement parameters: CSO2Lo and CSO2Hi.

VB Syntax *imd.CSOOffset = value*

Variable [\(Type\)](#) - Description

imd A [SweptIMD](#) Object

value (Double) Offset value in dBm.

Return Type Double

Default 0

Examples

```
imd.CSOOffset = 2 'Write  
value = imd.CSOOffset 'Read
```

C++ Syntax HRESULT get_CSOffset(double *pVal)
HRESULT put_CSOffset(double pVal)

Interface ISweptIMD

Last Modified:

16-Sep-2008 MX New topic

CTBOffset Property

Description Sets and returns the offset that is applied to CTB measurements. Valid only with measurement parameters: CTB, CTBLo, CTBHi, CTBE, CTBELo, and CTBEHi.

VB Syntax *imd.CTBOffset = value*

Variable [\(Type\)](#) - Description

imd A [SweptIMD](#) Object

value (Double) Offset value in dBm.

Return Type Double

Default 0

Examples

```
imd.CTBOffset = 2 'Write  
value = imd.CTBOffset 'Read
```

C++ Syntax HRESULT get_CTBOffset(double *pVal)
HRESULT put_CTBOffset(double pVal)

Interface ISweptIMD

Last Modified:

7-Apr-2009 Added valid parameters
16-Sep-2008 MX New topic

CTBXMODNumCarriers Property

Description Sets the “N = Total number of carriers” value used in the calculation of the XMOD and CTB parameter.
[Learn more.](#)

VB Syntax *imd.CTBXMODNumCarriers = value*

Variable [\(Type\)](#) - Description

imd A [SweptIMD](#) Object

value (Long Integer) Number of carriers.

Return Type Long Integer

Default 40

Examples

```
imd.CTBXMODNumCarriers = 15 'Write
```

```
value = imd.CTBXMODNumCarriers 'Read
```

C++ Syntax HRESULT get_CTBXMODNumCarriers(double *pVal)
HRESULT put_CTBXMODNumCarriers(double *pVal)

Interface ISweptIMD

Last Modified:

25-Aug-2008 MX New topic

CustomCalConfiguration Property

Description Calibration for the following PNA Applications is performed using the [GuidedCalibration Object](#). This command provides access to additional Properties and Methods which extends the GuidedCal Object.

Meas Type	Custom Cal Interface
Gain Compression	IGainCompressionCal Object
Noise Figure	INoiseCal Object
Swept IMD	ISweptIMD Cal Object

VB Syntax *set custCal = cal.***CustomCalConfiguration()**

Variable [\(Type\)](#) - Description

custCal **(object)** The handle to an interface that provides application-specific calibration properties..

cal [GuidedCalibration Object](#).

Return Type Depends on the MeasType. See above table.

Default None

Examples See examples: [NoiseFigure](#) [GainCompression](#) [SweptIMD](#)

C++ Syntax HRESULT CustomCalConfiguration(IDispatch** value);

Interface IGuidedCalibration4

Last Modified:

22-Jul-2009 Updated description

8-Sep-2008 MX New topic

CustomChannelConfiguration Property

Description Returns a handle to the custom application object on the active channel. You can either **(1)** use the handle directly to access measurement properties and methods, or **(2)** set a variable to the measurement object. The variable retains a handle to the original measurement.

Currently, the custom application objects to which this property provides access are:

- [NoiseFigure Object](#)
- [GainCompression Object](#)
- [SweptIMD Object](#)

VB Syntax 1) *set custChan = chan.CustomChannelConfiguration.* <setting>
or
2) *set custChan = app.ActiveChannel.CustomChannelConfiguration*

Variable [\(Type\)](#) - Description

custChan A variable in which the handle to a custom application is returned. **(object)**

chan A [Channel](#) **(object)**

<setting> A property or method setting on the custom application object.

Return Type Custom application object

Default None

Examples See examples: [NoiseFigure](#) [GainCompression](#) [SweptIMD](#)

C++ Syntax HRESULT CustomChannelConfiguration(IDispatch** value);

Interface IChannel12

Last Modified:

27-Aug-2008 Added Swept IMD
16-Oct-2007 MX New topic

CustomMeasurementConfiguration Property

Description Returns a handle to a custom measurement object on the active channel. You can use the handle to access custom measurement properties and methods.

Currently, the custom measurement objects to which this property provides access is:

- [GainCompressionMeas Object](#)

VB Syntax *Set custMeas = meas.***CustomMeasurementConfiguration**

Variable [\(Type\)](#) - **Description**

custMeas A variable in which the handle to a custom measurement is returned. **(object)**

chan A [Measurement](#) **(object)**

Return Type Custom Measurement object

Default None

Examples See examples: [GainCompression](#)

C++ Syntax HRESULT CustomMeasurementConfiguration(IDispatch** value);

Interface IMeasurement12

Last Modified:

3-Sep-2009 MX New topic

CWFrequency Property

Description Set the Continuous Wave (CW) frequency. Must first send `chan.SweepType = naCWTimeSweep`.

See Also: [calset.CWFrequency Property](#)

VB Syntax `object.CWFrequency = value`

Variable [\(Type\)](#) - Description

object One of the following:

- [Channel](#) (object)
- [FOMRange](#) (object) Range must be [UNCOUPLED](#).

See also [Measurement2](#) interface.

value **(double)** CW frequency. Choose any number between:
the **minimum** and **maximum** frequency limits of the analyzer
Units are Hz

Return Type Double

Default 1e9

Examples

```
chan.CWFrequency = 5e9 'Write
```

```
cwfreq = chan.CWFrequency 'Read
```

C++ Syntax HRESULT put_CWFrequency(double newVal)
HRESULT get_CWFrequency(double *pVal)

Interface IChannel

IFOMRange

CWFrequency (Cal Set) Property

Description Returns the CW frequency that is stored in the Cal Set.

VB Syntax *value* = *CalSet.CWFrequency* (*range*)

Variable [\(Type\)](#) - Description

value **(double)** - returned CW frequency in Hertz.

CalSet [CalSet](#) **(object)**

range **(Long)** Choose: **0**

Return Type Double

Default Not Applicable

Examples `cw = calset.CWFrequency(0) 'Reads the CW frequency stored in the cal set.`

C++ Syntax HRESULT get_CWFrequency(long range, double *pVal)

Interface |CalSet3

Last modified:

15-Feb- 2011 Added new topic

Data Property

Description Reads the next specified number of data points from the FIFO buffer. The data is returned in 32 bit real/imaginary pair. Data is cleared as it is read.

Note: This method is the slowest way to transfer data using COM. However, it is supported by all COM client programming languages. For better performance, try using [DataInCompactForm](#).

VB Syntax `array = fifo.Data (count)`

Variable [\(Type\) - Description](#)

`array` (Variant) Variable to store the returned data.

`fifo` A [FIFO](#) Object

`count` (Long Integer) Number of data points to read.

Return Type Variant array. Each VARIANT is typed as a 4-byte floating point number.

Default Not Applicable

Examples `value = fifo.Data(500) 'Read`

C++ Syntax `HRESULT get_Data(long count, VARIANT * data);`

Interface IFIFO

Last Modified:

7-Oct-2008 MX New topic

DataAndLimits Property

Description Set and return the color of Data and Limit Lines for nth trace in a window.

VB Syntax `trace(n).DataAndLimits = value`

Variable [\(Type\)](#) - Description

`trace(n)` One of the 8 [ComTraceColors](#) objects

`value` **(Long Integer)** - RGB color of the DataAndLimits pen.

Convert the three RGB colors to an integer as follows:

```
RGB = R+(G*2^8)+(B*2^16)
```

To find the three RGB values from the [Display Colors dialog](#), click **Change Color**, then **Define Custom Color**.

Return Type Long

Default Varies for each trace.

Examples

```
R = 10
G = 10
B = 10
RGB = 10+(10*2^8)+(10*2^16)
tracel.DataAndLimits = RGB 'Write
color = tracel.DataAndLimits 'Read
```

C++ Syntax HRESULT get_DataAndLimits(long* pVal);

HRESULT put_DataAndLimits(long newVal);

Interface IComTraceColors

Last Modified:

7-Aug-2009 MX New topic

DataCount Property

Description	Returns the total number of data points in the FIFO buffer.
VB Syntax	<i>value</i> = <i>fifo</i> . DataCount
Variable	(Type) - Description
<i>value</i>	(Long Integer) Variable to store the returned number of data points.
<i>fifo</i>	A FIFO Object
Return Type	Long Integer
Default	Not Applicable
Examples	<code>value = fifo.DataCount 'Read</code>
C++ Syntax	HRESULT get_DataCount(long *value)
Interface	IFIFO

Last Modified:

7-Oct-2008 MX New topic

DataInCompactForm Property

Description Reads FIFO data the same as [Data Property](#) but returns the data in a more compact form of SAFEARRAY. This is significantly faster but it is not supported in all client environments.

VB Syntax `array = fifo.DataInCompactForm (count)`

Variable [\(Type\)](#) - Description

`array` (Variant) Variable to store the returned data

`fifo` A [FIFO](#) Object

`count` (Long Integer) Number of data points to read.

Return Type Returns an array of 4 byte floating point numbers.

Default Not Applicable

Examples `value = fifo.DataInCompactForm(500) 'Read`

C++ Syntax HRESULT get_DataInCompactForm(long count, VARIANT * data);

Interface IFIFO

DCCorrection Property

Description Sets and returns the correction ON/OFF state for a DC Meter and a DC Source.

VB Syntax `extDC.DCCorrection = value`

Variable [\(Type\)](#) - Description

extDC An [ExternalDCDevice](#) (object)

value (**Boolean**) Correction ON/OFF state. Choose from:

True - Turn Correction ON

False - Turn Correction OFF

Return Type Boolean

Default False

Examples `extDC.DCCorrection = True 'Write`

`bool = extDC.DCCorrection 'Read`

C++ Syntax HRESULT get_DCCorrection (BOOL *pValue)

HRESULT put_DCCorrection (BOOL newVal)

Interface IExternalDCDevice

Last Modified:

15-Feb-2012 New topic

DOffset Property

Description Sets and returns the offset correction value for an external DC Device which can be configured as either a DC Meter or a DC Source.

VB Syntax *extDC.DOffset = value*

Variable [\(Type\)](#) - Description

extDC An [ExternalDCDevice](#) (**object**)

value (**Double**) DC offset value.

The PNA will display readings from a DC Meter as:

```
Display = (Meas'd value - Offset) * Scale
```

The PNA will adjust the output from a DC Source as:

```
Output = (Set value - Offset) * Scale
```

Return Type Double

Default 0

Examples `extDC.DOffset = 4 'Write`

```
offset = extDC.DOffset 'Read
```

C++ Syntax HRESULT get_DOffset (double *pValue)

HRESULT put_DOffset (double newVal)

Interface IExternalDCDevice

Last Modified:

15-Feb-2012 New topic

DCScale Property

Description Sets and returns the scale correction value for an external DC Device which can be configured as either a DC Meter or a DC Source.

VB Syntax *extDC.DCScale = value*

Variable [\(Type\)](#) - Description

extDC An [ExternalDCDevice](#) (**object**)

value (**Double**) DC scale value.

The PNA will display readings from a DC Meter as:

```
Display = (Meas'd value - Offset) * Scale
```

The PNA will adjust the output from a DC Source as:

```
Output = (Set value - Offset) * Scale
```

Return Type Double

Default 1

Examples `extDC.DCScale = -2 'Write`

```
slope = extDC.DCScale 'Read
```

C++ Syntax HRESULT get_DCScale (double *pValue)

HRESULT put_DCScale (double newVal)

Interface IExternalDCDevice

Last Modified:

15-Feb-2012 New topic

DCType Property

Description Sets and returns the DC Type for an external DC Device which can be configured as either a DC Meter or a DC Source. This setting is used as the units for display on the PNA X-axis.

VB Syntax `extDC.DCType = value`

Variable [\(Type\) - Description](#)

extDC An [ExternalDCDevice](#) (object)

value **(String)** DC type. Choose from:
"dBm", "A", "V", "W", "K", "F", "C"

Return Type String

Default "V"

Examples
`extDC.DCType = "A" 'Write`
`units = extDC.DCType 'Read`

C++ Syntax HRESULT get_DCType (BSTR *pValue)
HRESULT put_DCType (BSTR newVal)

Interface IExternalDCDevice

Last Modified:

15-Feb-2012 New topic

DefinedRoles Property

Description	This command replaces GetSourceRoles Method . Returns the roles for which sources can be used for the channel. Use RoleDevice to assign a source to a role.
VB Syntax	<i>value</i> = <i>chan</i> . DefinedRoles
Variable	(Type) - Description
	<i>value</i> (Variant) - Variable to store the returned channel roles.
	<i>chan</i> A Channel (object)
Return Type	Variant
Default	Not Applicable
Examples	<pre>vRoles = chan.DefinedRoles 'Read</pre>
C++ Syntax	HRESULT get_DefinedRoles(Variant* roles);
Interface	IChannel22

Last modified:

12-Mar-2014 New topic

Delay Property

Description Sets and Returns the electrical delay value for the calibration standard.

VB Syntax *calstd.Delay* = *value*

Variable [\(Type\)](#) - Description

calstd A CalStandard (**object**). Use calKit.[GetCalStandard](#) to get a handle to the standard.

value (**single**) - Electrical delay in picoseconds

Return Type Single

Default Not Applicable

Exaamples `calstd.Delay = 12 'Write 12ps Delay`
`stdDelay = calstd.Delay 'Read the value of Delay`

C++ Syntax HRESULT get_Delay(float *pVal)
HRESULT put_Delay(float newVal)

Interface ICalStandard

Last Modified:

5-Jan-2007 MX New topic

Delay (pulse) Property

Description Sets the pulse delay - the amount of time before a new pulse begins.

VB Syntax *pulse*.Delay(*n*) = *value*

Variable [\(Type\)](#) - Description

pulse A [PulseGenerator](#) (object)

n (**Integer**) Pulse generator number. Choose from 0 to 4.

Or use [PulseGeneratorID](#) to refer to an external pulse generator.

0 is the generator that pulses the ADC.

value (**Double**) Delay value in seconds. Choose a value from about 33ns to about 70 seconds.

Return Type Double

Default 0

Examples `pulse.Delay(1) = 1ms 'Write`

`value = pulse.Delay(4) 'Read`

C++ Syntax HRESULT get_Delay(integer pulse, double* delay);

HRESULT put_Delay(integer pulse, double delay);

Interface IPulseGenerator

Last Modified:

16-Feb-2012 Edit for ext PG

1-Jan-2007 MX New topic

Delay Property

Description Specifies the delay that should be applied by the PNA after the aux trigger input is received and before the acquisition is made.

Note: Use on PNA-X ONLY. Other models do NOT have an Aux Input.

VB Syntax `auxTrig.Delay = value`

Variable [\(Type\)](#) - Description

`auxTrig` An [AuxTrigger](#) (object)

`value` (**double**) - Delay value in seconds. Choose a value between 0 and 3.0 seconds.

Return Type Double

Default Not Applicable

Examples

```
auxTrig.Delay = 1.2 'Write 1.2s Delay
value = auxTrig.Delay 'Read the value
```

C++ Syntax HRESULT get_Delay(double *val);
HRESULT put_Delay(double val);

Interface IAuxTrigger

Last Modified:

5-Sep-2008 Added Note
14-Dec-2006 MX New topic

DelayCalculationMethod Property

Description Set and return the method of setting the delay through the calibration mixer.

To select Phase Reference Cal method for correcting an SMC+Phase measurement, use [ImportDataSet Method](#).

VB Syntax `smc.DelayCalculationMethod = value`

Variable [\(Type\)](#) - Description

`smc` An [SMCType](#) (object)

`value` (Enum as `NADelayCalculationMethod`)

0 - `naDelayCalculationMethod_FixedDelay` - use a known delay value set with [FixedDelay Property](#)

1 - `naDelayCalculationMethod_MixerCharacterizationFile` - use the S2P file set with [MixerCharacterizationFile Property](#)

Return Type Enum

Default **0 - `naDelayCalculationMethod_FixedDelay`**

Example `SMC.DelayCalculationMethod = naDelayCalculationMethod_FixedDelay`

C++ Syntax `HRESULT put_DelayCalculationMethod(tagNADelayCalculationMethod Value);`

`HRESULT get_DelayCalculationMethod(tagNADelayCalculationMethod* Value);`

Interface `SMCType5`

Last Modified:

25-Mar-2010 MX New topic

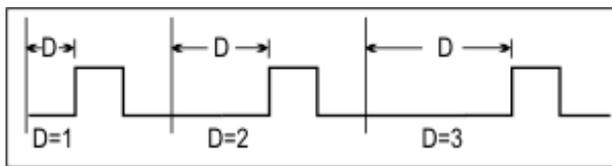
DelayIncrement Property

Description Sets the pulse delay increment. The delay increments with each pulse by the <value> amount.

For example, in this diagram the delay starts as 1. On the second pulse, delay=2. On the third pulse, delay=3.

Important: If $D + W$ is greater than P , then undefined PNA behavior results. There is NO error message or warning. Delay includes the incremented value.

This is useful for pulse profiling.



VB Syntax `pulse.DelayIncrement(n) = value`

Variable [\(Type\)](#) - Description

pulse A [PulseGenerator](#) (object)

n (**Integer**) Pulse generator number. Choose from 0 to 4.

Or use [PulseGeneratorID](#) to refer to an external pulse generator.

0 is the generator that pulses the ADC.

value (**Double**) Delay increment value in seconds.

Return Type Double

Default 0

Examples `pulse.DelayIncrement(1) = 1ms 'Write`

`value = pulse.DelayIncrement(4) 'Read`

C++ Syntax `HRESULT get_DelayIncrement(integer pulse, double* dIncr);`

`HRESULT put_DelayIncrement(integer pulse, double dIncr);`

Interface IPulseGenerator

16-Feb-2012 Edit for ext PG

18-Jun-2007 MX New topic

DeltaFrequency Property

Description Sets and returns the fixed tone spacing value. Use with IMD sweep types:

- `naIMDToneCWSweep`
- `naIMDTonePowerSweep`
- `naIMDToneCenterFreqSweep`

VB Syntax `object.DeltaFrequency = value`

Variable [\(Type\)](#) - Description

object A [SweptIMD](#) or [IMSpectrum](#) Object

value (Double) - Tone spacing frequency in Hz. Both the F1 and F2 tones MUST be within the frequency range of the PNA.

Return Type Double

Default 1 MHz

Examples

```
imd.DeltaFrequency = 1e6 'Write
```

```
value = imd.DeltaFrequency 'Read
```

C++ Syntax HRESULT get_DeltaFrequency(double *pVal)
HRESULT put_DeltaFrequency(double newVal)

Interface ISweptIMD
IIMSpectrum

Last Modified:

19-Aug-2008 MX New topic

DeltaFrequencyStart Property

Description Sets and returns the starting main tone separation for [sweep type](#)=naIMDDeltaFrequencySweep

VB Syntax *imd.DeltaFrequencyStart = value*

Variable [\(Type\)](#) - Description

imd A [SweptIMD](#) Object

value (Double) - Starting tone separation between F1 and F2 in Hz. Both F1 and F2 tones MUST be within the frequency range of the PNA where:

$F1 \text{ (start)} = imd.FrequencyCenter - imd.DeltaFrequencyStart / 2$

$F2 \text{ (start)} = imd.FrequencyCenter + imd.DeltaFrequencyStart / 2$

Return Type Double

Default 1 MHz

Examples

```
imd.DeltaFrequencyStart = 5e6 'Write
```

```
value = imd.DeltaFrequencyStart 'Read
```

C++ Syntax HRESULT get_DeltaFrequencyStart(double *pVal)
HRESULT put_DeltaFrequencyStart(double newVal)

Interface ISweptIMD

Last Modified:

19-Aug-2008 MX New topic

DeltaFrequencyStop Property

Description Sets and returns the stop spacing of the main tones. Use with [sweep type](#)=naIMDDeltaFrequencySweep

VB Syntax *imd.DeltaFrequencyStop = value*

Variable [\(Type\)](#) - Description

imd A [SweptIMD](#) Object

value (Double) - Stopping tone separation between F1 and F2 in Hz. Both F1 and F2 tones MUST be within the frequency range of the PNA where:

$F1 \text{ (stop)} = \text{imd.FrequencyCenter} - \text{imd.DeltaFrequencyStop} / 2$

$F2 \text{ (stop)} = \text{imd.FrequencyCenter} + \text{imd.DeltaFrequencyStop} / 2$

Return Type Double

Default 10 MHz

Examples

```
imd.DeltaFrequencyStop = 20e6 'Write
```

```
value = imd.DeltaFrequencyStop 'Read
```

C++ Syntax HRESULT get_DeltaFrequencyStop(double *pVal)
HRESULT put_DeltaFrequencyStop(double newVal)

Interface ISweptIMD

Last Modified:

19-Aug-2008 MX New topic

DeltaMarker Property

Description Sets a marker as a delta marker. The reference marker must already be turned ON. See [meas.ReferenceMarkerState](#)

VB Syntax `mark.DeltaMarker = state`

Variable [\(Type\)](#) - Description

mark A [Marker](#) (object)

state (boolean) -

True - marker is a delta marker

False - marker is NOT a delta marker

Return Type Boolean

Default False

Examples `mark.DeltaMarker = True 'Write`

`delta = mark.DeltaMarker 'Read`

C++ Syntax HRESULT get_DeltaMarker(VARIANT_BOOL bState)
HRESULT put_DeltaMarker(VARIANT_BOOL *bState)

Interface IMarker

Description Property

Description	Sets or returns the descriptive string assigned to the Cal Set. Change this string so that you can easily identify each Cal Set constructed.
VB Syntax	<i>CalSet</i> . Description = <i>value</i>
Variable	(Type) - Description
<i>CalSet</i>	(object) - A Cal Set object
<i>value</i>	(string) – Description of the Cal Set
Return Type	String
Default	“CalSet_n” where n is an integer number.
Examples	<pre>CalSet.Description = "My Cal Set" 'Write desc = CalSet.Description 'Read</pre>
C++ Syntax	HRESULT get_Description(BSTR *pVal) HRESULT put_Description(BSTR newVal);
Interface	ICalSet

Last modified:

Dec.12, 2006 MX New topic

DescriptiveText Property

Description	Write and read descriptive text associated with the configuration. This text is displayed in the path configuration dialog. Text is generally used to describe external connections that must be made manually to complete the configuration setup.
VB Syntax	<i>pathConfig</i> . DescriptiveText = <i>text</i>
Variable	(Type) - Description
<i>name</i>	(String) Variable to store the returned configuration name.
<i>pathConfig</i>	A PathConfiguration (object)
<i>text</i>	(String) Descriptive text enclosed in quotes.
Return Type	String
Default	Not Applicable
Examples	<pre>pathConf.DescriptiveText "here are the instructions for connecting the device for this configuration"</pre>
C++ Syntax	<pre>HRESULT get_DescriptionText(BSTR* pConnectionText); HRESULT put_DescriptionText(BSTR connectionText);</pre>
Interface	IPathConfiguration

Last Modified:

14-Dec-2006 MX New topic

DeviceInputPort Property

Description Returns the PNA port number to be used for the DUT input.

VB Syntax *port* = *obj*.**DeviceInputPort**

Variable [\(Type\)](#) - Description

port **(long)** Variable to store the PNA port number of the DUT input.

mixer A [Mixer](#) **(object)**

Return Type Long

Default 1

Examples `value = mixer.DeviceInputPort`

C++ Syntax HRESULT get_DeviceInputPort(long *pVal)

Interface IMixer8

Last Modified:

23-Apr-2008 New topic.

DeviceInputPort Property

Description Read the PNA port number which is connected to the DUT input.

Use [SetPortMap Method](#) to change the port mapping.

VB Syntax *obj*.DeviceInputPort

Variable [\(Type\)](#) - Description

obj A [Converter](#) (**object**) or
A [GainCompression](#) (**object**) or
A [SweptIMD](#) (**object**) or
An [IMSpectrum](#) (**object**)
A [NoiseFigure](#) (**object**)

Return Type Integer

Default 1

Examples `inPort = gca.DeviceInputPort 'Read`

C++ Syntax HRESULT get_DeviceInputPort(int* pVal)

Interface IGainCompression

ISweptIMD

IMSpectrum

INoiseFigure6

IConverter6

Last Modified:

23-May-2011	Added converter
30-Apr-2010	Added NF
11-Aug-2009	Added IMD and IMS (9.0)
11-Sep-2007	MX New topic

DeviceNames Property

Description	Returns a list of configured device names of the specified device type.
VB Syntax	<i>value</i> = <i>extDevices</i> . DeviceNames (<i>devType</i>)
Variable	(Type) - Description
<i>value</i>	(Variant) Variable to store the device names.
<i>extDevices</i>	An ExternalDevices (collection)
<i>devType</i>	(String) Device Type such as "Source" and "Power Meter". See a complete list of valid device types .
Return Type	Variant Array
Default	Not Applicable
Examples	<pre>devices = extDevices.DeviceNames("Power Meter") 'Read</pre>
C++ Syntax	HRESULT get_DeviceNames(VARIANT *names);
Interface	IExternalDevices2

Last Modified:

19-Jul-2011 New topic

DeviceOutputPort Property

Description Returns the PNA port number to be used for the DUT output.

VB Syntax *port* = *mixer*.**DeviceOutputPort**

Variable [\(Type\)](#) - Description

port **(long)** Variable to store the PNA port number of the DUT output.

mixer A [Mixer](#) **(object)**

Return Type Long

Default 2

Examples `value = mixer.DeviceOutputPort`

C++ Syntax HRESULT get_DeviceOutputPort(long *pVal)

Interface IMixer8

Last Modified:

23-Apr-2008 New topic.

InputLinearPowerLevel Property

Description Set and read the input power at which Linear Gain and all S-parameters are measured.

VB Syntax `gca.InputLinearPowerLevel = value`

Variable [\(Type\)](#) - Description

gca A [GainCompression](#) (object)

value (**double**) Linear input power level in dBm. Choose a value from +30 to (-30).

Return Type Double

Default - 25 dBm

Examples `gca.InputLinearPowerLevel = -10 'Write`

`LinPwr = gca.InputLinearPowerLevel 'Read`

C++ Syntax HRESULT get_InputLinearPowerLevel(double* pVal)

HRESULT put_InputLinearPowerLevel(double newVal)

Interface IGainCompression

Last Modified:

11-Sep-2007 MX New topic

DeviceOutputPort Property

Description Read the PNA port number which is connected to the DUT Output.

Use [SetPortMap Method](#) to change the port mapping.

VB Syntax *obj*.DeviceOutputPort

Variable (Type) - Description

obj A [Converter](#) (object) or

A [GainCompression](#) (object) or

A [SweptIMD](#) (object) or

An [IMSpectrum](#) (object)

A [NoiseFigure](#) (object)

Return Type Integer

Default 2

Examples `outPort = gca.DeviceOutputPort 'Read`

C++ Syntax HRESULT get_DeviceOutputPort(int* pVal)

Interface IGainCompression

ISweptIMD

IMSpectrum

INoiseFigure6

IConverter6

Last Modified:

23-May-2011	Added Converter
30-Apr-2010	Added NF
11-Aug-2009	Added IMD and IMS
11-Sep-2007	MX New topic

DeviceType Property

Description Sets and returns the DeviceType for the external device.

VB Syntax *extDevices.DeviceType = value*

Variable [\(Type\)](#) - Description

extDevices An [ExternalDevice](#) (object)

value (String) Device Type. Choose from:

"Source" for external source.

"Power Meter" for power meter.

Return Type String

Default "Unknown"

Examples

```
extDevices.DeviceType = "source" Write
```

```
value = extDevices.DeviceType 'Read
```

```
See example program to configure PMAR device
```

```
See example program to configure External Source
```

C++ Syntax HRESULT get_DeviceType(BSTR* value);

HRESULT put_DeviceType(BSTR newVal);

Interface IExternalDevices

Last Modified:

31-Jul-2009 MX New topic

DiffPortMatch_C Property

Description Sets the Capacitance value of the differential matching circuit.

VB Syntax *fixture*.DiffPortMatch_C (*portNum*) = *value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

portNum **(Integer)** Balanced (logical) port number. Choose from logical ports 1,2 ,3. [Learn more about logical ports.](#)

value **(Double)** Capacitance value in farads. Choose a value between **-1E18** to **1E18**.

Return Type Double

Default 0

Examples `fixture.DiffPortMatch_C(2) = 1e-6 'Write`

`value = fixture.DiffPortMatch_C(1) 'Read`

C++ Syntax HRESULT get_DiffPortMatch_C(short portNum, double *pVal)
HRESULT put_DiffPortMatch_C(short portNum, double newVal)

Interface IFixturing2

DiffPortMatch_G Property

Description Sets the Conductance value of the differential matching circuit.

VB Syntax *fixture*.DiffPortMatch_G(*portNum*) = *value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

portNum **(Integer)** Balanced (logical) port number. Choose from logical ports 1, 2, or 3. [Learn more about logical ports.](#)

value **(Double)** Conductance value in siemens. Choose a value between **-1E18** to **1E18**.

Return Type Double

Default 0

Examples `fixture.DiffPortMatch_G(2) = 1e-3 'Write`

`value = fixture.DiffPortMatch_G(1) 'Read`

C++ Syntax HRESULT get_DiffPortMatch_G(short portNum, double *pVal)
HRESULT put_DiffPortMatch_G(short portNum, double newVal)

Interface IFixturing2

DiffPortMatch_L Property

Description Sets the Inductance value of the differential matching circuit.

VB Syntax *fixture.DiffPortMatch_L(portNum) = value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

portNum **(Integer)** Balanced (logical) port number. Choose from logical ports 1, 2, or 3. [Learn more about logical ports.](#)

value **(Double)** Inductance value in henries. Choose a value between **-1E18** to **1E18**.

Return Type Double

Default 0

Examples `fixture.DiffPortMatch_L(2) = 1e-3 'Write`

`value = fixture.DiffPortMatch_L(1) 'Read`

C++ Syntax HRESULT get_DiffPortMatch_L(short portNum, double *pVal)
HRESULT put_DiffPortMatch_L(short portNum, double newVal)

Interface IFixturing2

DiffPortMatch_R Property

Description Sets the Resistance value of the differential matching circuit.

VB Syntax `fixture.DiffPortMatch_R(portNum) = value`

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

portNum **(Integer)** Balanced (logical) port number. Choose from logical ports 1, 2, or 3. [Learn more about logical ports.](#)

value **(Double)** Resistance value in ohms. Choose a value between **-1E18** to **1E18**.

Return Type Double

Default 0

Examples `fixture.DiffPortMatch_R(2) = 1e3 'Write`

`value = fixture.DiffPortMatch_R(1) 'Read`

C++ Syntax HRESULT get_DiffPortMatch_R(short portNum, double *pVal)
HRESULT put_DiffPortMatch_R(short portNum, double newVal)

Interface IFixturing2

DiffPortMatchMode Property

Description Sets the differential matching circuit type. To select a user-defined circuit, specify IN ADVANCE the 2-port touchstone filename with [DiffPortMatch_UserFilename Property](#). If you do not specify the appropriate file and you select USER, an error occurs and naNO_CIRCUIT is automatically selected.

VB Syntax *fixture*.DiffPortMatchMode(*pNum*) = *value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

pNum (**Integer**) Balanced (logical) port number. Choose from logical ports **1, 2, or 3**. [Learn more about logical ports](#).

value (**Enum as NADiffPortMatchCircuitMode**) Choose from:

0 or **naSHUNT_L_SHUNT_C_CIRCUIT** - Specifies the circuit that consists of shunt L and shunt C.

1 or **naUSER_FILE_CIRCUIT** - Specifies the user-defined circuit.

2 or **naNO_CIRCUIT** - Specifies no-circuit.

Return Type Enum

Default **naSHUNT_L_SHUNT_C_CIRCUIT**

Examples

```
fixture.DiffPortMatchMode(2) = naNO_CIRCUIT 'Write
value = fixture.DiffPortMatchMode(1) 'Read
```

C++ Syntax HRESULT get_DiffPortMatchMode(short port, tagNADiffPortMatchCircuitMode *eVal)

HRESULT put_DiffPortMatchMode(short port, tagNADiffPortMatchCircuitMode eVal)

Interface IFixturing2

DiffPortMatchState Property

Description Turns ON or OFF 4-port differential port matching function. Must also set the fixture simulator function to ON using [FixturingState Property](#).

VB Syntax *fixture.DiffPortMatchState = value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

value (Boolean)

False - Turns differential port matching OFF

True - Turns differential port matching ON

Return Type Boolean

Default False

Examples

```
fixture.DiffPortMatchState = False 'Write
```

```
value = fixture.DiffPortMatchState 'Read
```

C++ Syntax HRESULT get_DiffPortMatchState(VARIANT_BOOL *pVal)
HRESULT put_DiffPortMatchState(VARIANT_BOOL newVal)

Interface IFixturing2

DiffPortMatchUserFilename Property

Description Specifies the 2-port touchstone file in which the information on the user-defined differential matching circuit is saved. Following this command, send [DiffPortMatchCircuit Property](#). If the specified file does not exist, an error occurs when you set the type of differential matching circuit to USER.

VB Syntax *fixture.DiffPortMatchUserFilename(pNum) = value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

pNum (**Integer**) Balanced (logical) port number. Choose from logical ports **1, 2** or **3**. [Learn more about logical ports.](#)

value (**String**) Full path, file name, and extension (.s2P) of the de-embedding circuit.

Files are typically stored in "c:\users\public\network analyzer\documents

Return Type String

Default Not Applicable

Examples

```
fixture.DiffPortMatchUserFilename(2) = "c:\users\public\network analyzer\documents/myFile.s4p" 'Write
```

```
value = fixture.DiffPortMatchUserFilename(1) 'Read
```

C++ Syntax HRESULT get_DiffPortMatchUserFilename(short port, BSTR *bstrFile)

HRESULT put_DiffPortMatchUserFilename(short port, BSTR bstrFile)

Interface IFixturing2

DiffZConvPortImag Property

Description Sets the imaginary part of the impedance value for the differential port impedance conversion function.

VB Syntax `fixture.DiffZConvPortImag(portNum) = value`

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

portNum **(Integer)** Balanced (logical) port number. Choose from logical ports 1, 2, or 3. [Learn more about logical ports.](#)

value **(Double)** Imaginary part of the Impedance value. Choose a value between 0 and 1E18

Return Type Double

Default 0

Examples `fixture.DiffZConvPortImag(2) = 75 'Write`

`value = fixture.DiffZConvPortImag(1) 'Read`

C++ Syntax HRESULT get_DiffZConvPortImag(short portNum, double *pVal)
HRESULT put_DiffZConvPortImag(short portNum, double newVal)

Interface IFixturing2

DiffZConvPortReal Property

Description Sets the real part of the impedance value for the differential port impedance conversion function.

VB Syntax `fixture.DiffZConvPortReal(portNum) = value`

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

portNum **(Integer)** Balanced (logical) port number. Choose from logical ports 1, 2, or 3. [Learn more about logical ports.](#)

value **(Double)** Real part of the Impedance value. Choose a value between 0 and 1E18

Return Type Double

Default See [Differential Port Z Conversion Default](#)

Examples

```
fixture.DiffZConvPortReal(2) = 75 'Write
```

```
value = fixture.DiffZConvPortReal(1) 'Read
```

C++ Syntax HRESULT get_DiffZConvPortReal(short portNum, double *pVal)
HRESULT put_DiffZConvPortReal(short portNum, double newVal)

Interface IFixturing2

Last Modified:

31-May-2013 Fixed typo

DiffZConvPortZ0 Property

Description Sets the impedance value for the differential port impedance conversion function. Set either this single value or set the [real](#) and [imaginary](#) parts separately. The imaginary part is set to 0.0 using this command.

VB Syntax `fixture.DiffZConvPortZ0(portNum) = value`

Variable [\(Type\) - Description](#)

fixture A [Fixturing](#) (object)

portNum (**Integer**) Balanced (logical) port number. Choose from logical ports 1, 2, or 3. [Learn more about logical ports.](#)

value (**Double**) Impedance value. Choose a value between 0 and 1E18

Return Type Double

Default See [Differential Port Z Conversion Default](#)

Examples

```
fixture.DiffZConvPortZ0(2) = 75 'Write
value = fixture.DiffZConvPortZ0(1) 'Read
```

C++ Syntax HRESULT get_DiffZConvPortZ0(short portNum, double *pVal)
HRESULT put_DiffZConvPortZ0(short portNum, double newVal)

Interface IFixturing2

DiffZConvState Property

Description Turns ON or OFF 4-port differential impedance conversion function. Must also set the fixture simulator function to ON using [FixturingState Property](#).

VB Syntax `fixture.DiffZConvState = value`

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

value (Boolean)

False - Turns differential impedance conversion OFF.

True - Turns differential impedance conversion ON.

Return Type Boolean

Default False

Examples

```
fixture.DiffZConvState = False 'Write  
value = fixture.DiffZConvState 'Read
```

C++ Syntax HRESULT get_DiffZConvState(VARIANT_BOOL *pVal)
HRESULT put_DiffZConvState(VARIANT_BOOL newVal)

Interface IFixturing2

Format Property

Description Sets or returns the display format of the measurement.

VB Syntax *meas.Format = value*

Variable [\(Type\)](#) - Description

meas A Measurement (**object**)

value (**enum NADataFormat**) - Choose from:

0 - naDataFormat_LinMag

1 - naDataFormat_LogMag

2 - naDataFormat_Phase

3 - naDataFormat_Polar

4 - naDataFormat_Smith

5 - naDataFormat_Delay

6 - naDataFormat_Real

7 - naDataFormat_Imaginary

8 - naDataFormat_SWR

9 - naDataFormat_PhaseUnwrapped

10 - naDataFormat_InverseSmith

11 - naDataFormat_Kelvin

12 - naDataFormat_Fahrenheit

13 - naDataFormat_Celsius

Return Type Long Integer

Default 1 - naDataFormat_LogMag

Examples

```
meas.Format = naDataFormat_Real 'Write
```

```
fmt = meas.Format 'Read
```

C++ Syntax HRESULT get_Format(tagDataFormat *pVal)
HRESULT put_Format(tagDataFormat newVal)
Interface IMeasurement

Last Modified:

1-Oct-2007 Added temperature formats

DisplayAutomationErrors Property

Description Enables or disables automation error messages from being displayed on the screen.

VB Syntax `app.DisplayAutomationErrors = value`

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

value (Boolean)

True allows error to show on display,

False turns error off from display.

Return Type Boolean

Default True

Examples

```
Dim app As Application
Set app = New Application
app.DisplayAutomationErrors = False      'Turns off display
print app.DisplayAutomationErrors       'prints False
```

C++ Syntax HRESULT get_DisplayAutomationErrors(VARIANT_BOOL * Val);

HRESULT put_DisplayAutomationErrors(VARIANT_BOOL Val);

Interface IApplication2

DisplayGlobalPassFail Property

Description Shows or hides the dialog which displays global pass/fail results. [Learn more about Global Pass/Fail.](#)

VB Syntax `app.DisplayGlobalPassFail = value`

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

value (Boolean)

True - displays the pass/fail dialog.

False - hides the pass/fail dialog.

Return Type Boolean

Default False

Examples

```
Dim app As Application
Set app = New Application
app.DisplayGlobalPassFail = true 'shows dialog
```

C++ Syntax HRESULT get_DisplayGlobalPassFail(VARIANT_BOOL * Val);

HRESULT put_DisplayGlobalPassFail(VARIANT_BOOL Val);

Interface IApplication6

DisplayRange Property

Description	Sets or returns the range to be displayed on the PNA x-axis. All traces in the channel have this same x-axis scaling.
VB Syntax	<i>FOM.DisplayRange = value</i>
Variable	(Type) - Description
<i>object</i>	An FOM (object)
<i>value</i>	(string) - Range to be displayed on the PNA x-axis. Case insensitive.
Return Type	String
Default	"Receivers"
Examples	<pre>fom.DisplayRange = "Source" 'sets the x-axis to the frequency range of "source"</pre> <pre>disprange = fom.DisplayRange 'Read</pre>
C++ Syntax	HRESULT get_DisplayRange(BSTR *pDspRange) HRESULT put_DisplayRange(BSTR pDspRange)
Interface	IFOM

Last Modified:

7-Mar-2007 Changed to receivers

Distance Property

Description Set or query marker distance on a time domain trace.

The Write command moves the marker to the specified distance value. Once moved, you can read the [Y axis](#) value or [read the X-axis time](#) value. (Distance is calculated from the X-axis time value.)

The Read command reads the distance of the marker.

If the marker is set as delta, the WRITE and READ data is relative to the reference marker.

VB Syntax *mark.Distance = value*

Variable [\(Type\)](#) - Description

mark A [Marker](#) (object)

value (**double**) - Marker distance in the unit of measure specified with [DistanceMarkerUnit Property](#)

Return Type Double

Default Not Applicable

Examples

```
mark.Distance = 3e9 'Write
xVal = mark.Distance 'Read
```

C++ Syntax HRESULT get_Distance(double *pVal);
HRESULT put_Distance(double newVal);

Interface IMarker2

DistanceMarkerMode Property

Description Specifies the measurement type in order to determine the correct marker distance.

- Select Auto for S-Parameter measurements.
- Select Reflection or Transmission for arbitrary ratio or unratiod measurements.

This settings affects the display of ALL markers for only the ACTIVE measurement.

VB Syntax *trans*.DistanceMarkerMode = *value*

Variable [\(Type\)](#) - [Description](#)

trans A [Transform](#) (object)

value (enum As NADistanceMarkerMode) - Choose from:

0 - naDistanceMarkerModeAuto

1 - naDistanceMarkerModeReflection

2 - naDistanceMarkerModeTransmission

Return Type Enum

Default 0 - naDistanceMarkerModeAuto

Examples `trans.DistanceMarkerMode = naDistanceMarkerModeReflection 'Write`

`DMM = trans.DistanceMarkerMode 'Read`

C++ Syntax HRESULT get_DistanceMarkerMode(tagNADistanceMarkerMode *pVal);

HRESULT put_DistanceMarkerMode(tagNADistanceMarkerMode newVal);

Interface ITransform2

DistanceMarkerUnit Property

Description	Specifies the unit of measure for the display of marker distance values. This settings affects the display of ALL markers for only the ACTIVE measurement (unless Distance Maker Units are coupled using CoupledParameters Property).
VB Syntax	<i>trans</i> . DistanceMarkerUnit = <i>value</i>
Variable	(Type) - Description
<i>trans</i>	A Transform (object)
<i>value</i>	(Enum As NADistanceMarkerUnit) - Distance Marker Units. Choose from 0 - naDistanceMarkerUnitMeter 1 - naDistanceMarkerUnitFeet 2 - naDistanceMarkerUnitInch
Return Type	Enum
Default	0 - naDistanceMarkerUnitMeter
Examples	<pre>trans.DistanceMarkerUnit = naDistanceMarkerUnitFeet 'sets the U = trans.DistanceMarkerUnit 'Read</pre>
C++ Syntax	HRESULT get_DistanceMarkerUnit(tagNADistanceMarkerUnit *pVal); HRESULT put_DistanceMarkerUnit(tagNADistanceMarkerUnit newVal);
Interface	ITransform2

Divisor Property

Description	Sets and returns the Divisor value to be used when coupling this range to the primary range. This setting is valid only if the specified range is coupled to the primary range.
VB Syntax	<i>FOMRange</i> . Divisor = <i>value</i>
Variable	(Type) - Description
<i>object</i>	An FOMRange (object)
<i>value</i>	(Double) - Divisor value.-(Unitless)
Return Type	Double
Default	0
Examples	<pre>fomRange.Divisor = .5 'Write</pre> <pre>Div = fomRange.Divisor 'Read</pre>
C++ Syntax	HRESULT get_Divisor(double *pVal) HRESULT put_Divisor(double *pVal)
Interface	IFOMRange

Do1PortEcal Property - Superseded

Description	Note: This command is replaced with CalKitType which sets the ECal module or mechanical cal kit. Specify ECAL or Mechanical calibration for the mixer characterization portion of a VMC calibration.
VB Syntax	<code>VMC.Do1PortEcal = bool</code>
Variable	(Type) - Description
<code>VMC</code>	VMCType (object)
<code>bool</code>	(Boolean)
	True - ECAL
	False - Mechanical
Return Type	Boolean
Default	False
Examples	<code>value = VMC.Do1PortEcal</code>
C++ Syntax	<code>HRESULT put_Do1PortEcal(VARIANT_BOOL bDoEcal);</code> <code>HRESULT get_Do1PortEcal(VARIANT_BOOL *bDoEcal);</code>
Interface	VMCType

Do2PortEcal Property - Superseded

Description	Note: This command is replaced with CalKitType which sets the ECal module or mechanical cal kit. Specify ECAL or Mechanical calibration. For VMC, this selection only applies to the 2-port calibration portion. For mixer characterization (VMC), use Do1PortEcal Property
VB Syntax	<i>object</i> . Do2PortEcal = <i>bool</i>
Variable	(Type) - Description
<i>object</i>	SMCType (object) or VMCType (object)
<i>bool</i>	(Boolean) True - ECAL False - Mechanical
Return Type	Boolean
Default	False
Examples	<code>value = VMC.Do2PortEcal</code>
C++ Syntax	HRESULT put_Do2PortEcal(VARIANT_BOOL bDoEcal); HRESULT get_Do2PortEcal(VARIANT_BOOL *bDoEcal);
Interface	SMCType VMCType

Domain Property

Description	Returns the domain (frequency,time, power, phase) of the measurement. To understand how this property is useful, see IMeasurement2 Interface .
VB Syntax	<i>value</i> = <i>meas</i> . Domain
Variable	(Type) - Description
<i>value</i>	(Enum as NADomainType) - variable to store the returned value
	0 - naDomainFrequency
	1 - naDomainTime
	2 - naDomainPower
	4 - naDomainPhase
<i>meas</i>	A Measurement (object)
Return Type	Enum as NADomainType
Default	Not Applicable
Examples	<pre>Print meas.Domain 'prints the value of the domain enum</pre>
C++ Syntax	HRESULT get_Domain(tagNADomainType * Val);
Interface	IMeasurement2

Last Modified:

3-May-2011 Updated for phase

Driver Property

Description Sets and returns the external device driver (model).

VB Syntax *extDevices.Driver* = *value*

Variable [\(Type\)](#) - Description

extDevices An [ExternalDevice](#) (object)

value (String) External device driver (model). Choose from the following:

"AGPM" for all power meters.

[See a list of supported external source drivers.](#)

Return Type String

Default "AGGeneric"

Examples

```
extDevices.Driver = "AGESG" 'Write
value = extDevices.Driver 'Read
See example program to configure PMAR device
See example program to configure External Source
```

C++ Syntax HRESULT get_Driver(BSTR* value);

HRESULT put_Driver(BSTR newVal);

Interface IExternalDevices

Last Modified:

15-Jan-2010 Added 'AG' to example

31-Jul-2009 MX New topic

DspFpgaRevision Property

Description	Returns the DSP FPGA Revision number that is visible in the Help, About Network Analyzer dialog box. Learn more.
VB Syntax	<i>value</i> = <i>cap</i> . DspFpgaRevision
Variable	(Type) - Description
<i>value</i>	(String) - Variable to store the returned DSP revision number.
<i>cap</i>	A Capabilities (object)
Return Type	String
Default	Not Applicable
Examples	<pre>value = cap.DspFpgaRevision 'Read</pre>
C++ Syntax	HRESULT get_DspFpgaRevision(BSTR *value);
Interface	ICapabilities6

Last Modified:

30-Nov-2009 MX New topic

DspRevision Property

Description	Returns the DSP Revision number that is visible in the Help, About Network Analyzer dialog box. Learn more.
VB Syntax	<i>value</i> = <i>cap</i> . DspRevision
Variable	(Type) - Description
<i>value</i>	(String) - Variable to store the returned DSP revision number.
<i>cap</i>	A Capabilities (object)
Return Type	String
Default	Not Applicable
Examples	<code>value = cap.DspRevision 'Read</code>
C++ Syntax	HRESULT get_DspRevision(BSTR *value);
Interface	ICapabilities6

DUTTopology Property

Description Returns the device topology setting.

VB Syntax *balTopology.DUTTopology = value*

Variable [\(Type\)](#) - Description

balTopology A [BalancedTopology](#) (object)

value (enum NADUTTopology) - Choose either:

0 naSEBal: Single-Ended - Balanced topology

1 naSESEBal: Single-Ended - Single-Ended - Balanced topology

2 naBalBal: Balanced - Balanced topology

3 naBalSE: Balanced - Single-ended topology

Return Type Enum as NADUTTopology

Default naSEBal

Examples `balTop.DUTTopology = naSESEBal 'Write`

`DutTop = balTop.DUTTopology 'Read`

C++ Syntax HRESULT get_DUTTopology(tagNADUTTopology* pVal)
HRESULT put_DUTTopology(tagNADUTTopology newVal)

Interface IBalancedTopology

Last Modified:

25-Apr-2012 Added Bal-Se topo (9.70)

DwellTime Property

Description Sets or returns the dwell time at the start of each sweep point for all measurements in a channel. Dwell time is only available with Chan.[SweepGenerationMode](#) = **naSteppedSweep** (not naAnalogSweep).

Sets or returns the dwell time of a specified sweep segment.

VB Syntax *object.DwellTime* = *value*

Variable [\(Type\)](#) - Description

object A Channel (object)

or CalSet (object) - Read-only property

or Segment (object)

value **(double)** - Dwell Time in seconds. Choose any number between **0** and **86400**

Return Type Double

Default 0

Examples

```
chan.DwellTime = 3e-3 'sets the dwell time for the channel -Write
```

```
segs(3).DwellTime = 1e9 'sets the dwell time of segment 3 -Write
```

```
dwell = chan.DwellTime 'Read
```

C++ Syntax HRESULT get_DwellTime(double *pVal)
HRESULT put_DwellTime(double newVal)

Interface IChannel
ISegment

|CalSet3

DwellPerPoint Property

Description Sets and returns the amount of time the PNA should wait after for an external source to settle before making a measurement at each data point.

VB Syntax *extSource.DwellPerPoint = value*

Variable [\(Type\)](#) - Description

chan An [ExternalSource](#) (**object**)

value (**Double**) Dwell time in milliseconds.

Return Type Double

Default 3

Examples `extSource.DwellPerPoint = 10 'Write`

`dpp = extSource.DwellPerPoint 'Read`

C++ Syntax HRESULT get_DwellPerPoint (tagNAExtDevDwellPerPoint *pValue)
HRESULT put_DwellPerPoint (tagNAExtDevDwellPerPoint newVal)

Interface IExternalSource

ECALCharacterization Property - Superseded

Description	<p>Note: This command is replaced with CalKitType which sets the ECal module and User Characterization.</p> <p>Specifies the characterization data within an ECal module to be used for the SMC calibration.</p> <p>Learn more about ECal User Characterization.</p>
VB Syntax	<code>SMC.ECALCharacterization(mod) = value</code>
Variable	(Type) - Description
<i>SMC</i>	SMCType (object)
<i>module</i>	1- ECal module
<i>value</i>	<p>(Long) – Characterization data within the ECal module to be used for ECal operations. Choose from:</p> <ul style="list-style-type: none"> 0 – Factory Characterization 1 – UserCharacterization1 2 – UserCharacterization2 3 – UserCharacterization3 4 – UserCharacterization4 5 – UserCharacterization5
Return Type	Long
Default	0 - Factory Characterization
Examples	<code>SMC.ECALCharacterization(1)= 2</code>
C++ Syntax	<pre>HRESULT put_ECALCharacterization(long moduleNumber, long characterization); HRESULT get_ECALCharacterization(long moduleNumber, long* characterization);</pre>
Interface	ISMCType

ECALCharacterization Property - Superseded

Description	<p>Note: This command is replaced with CalKitType which sets the ECal module and User Characterization.</p> <p>Specifies the characterization data within an ECal module to be used, and the portion of the VMC calibration.</p> <p>Learn more about ECal User Characterization.</p>
VB Syntax	<code>VMC.ECALCharacterization (module,port) = value</code>
Variable	(Type) - Description
<i>VMC</i>	VMCType (object)
<i>module</i>	(long integer)
	1 - ECAL module
<i>port</i>	(boolean)
	True - 2-port calibration portion of the VMC
	False - 1-port (mixer characterization portion of the VMC cal)
<i>value</i>	(Long) – Characterization data within the ECal module to be used for ECal operations. Choose from:
	0 – Factory Characterization
	1 – UserCharacterization1
	2 – UserCharacterization2
	3 – UserCharacterization3
	4 – UserCharacterization4
	5 – UserCharacterization5
Return Type	Long
Default	0 - Factory Characterization
Examples	<code>VMC.ECALCharacterization (1,True) = 4</code>

C++ Syntax HRESULT put_ECALCharacterization(long moduleNumber, long characterization);

HRESULT get_ECALCharacterization(long moduleNumber, long* characterization);

Interface IVMCType

ECALCharacterizationEx Property

Description	<p>This property replaces ECALCharacterization Property.</p> <p>Specifies the characterization data within an ECal module to be used for the calibration.</p> <p>Learn more about ECal User Characterization.</p>
VB Syntax	<i>cal</i> . ECALCharacterizationEx (<i>module</i>) = <i>value</i>
Variable	(Type) - Description
<i>cal</i>	Calibrator (object)
<i>module</i>	(long integer) Optional argument. ECal module.
	Choose from modules 1 through 8
	Use IsECALModuleFoundEx to determine the number of modules connected to the PNA
	Use GetECALModuleInfoEx to returns the model and serial number of each module.
<i>value</i>	<p>(Long) – Characterization data within the ECal module to be used for ECal operations. Choose from:</p> <p>0 – Factory Characterization</p> <p>1 – UserCharacterization1</p> <p>2 – UserCharacterization2</p> <p>..and so forth up to...</p> <p>12 – UserCharacterization12</p>
Return Type	Long
Default	0 - Factory Characterization
Examples	<code>cal.ECALCharacterizationEx (4) = 2</code>
C++ Syntax	<pre>HRESULT put_ECALCharacterizationEx(long moduleNumber, long characterization); HRESULT get_ECALCharacterizationEx(long moduleNumber, long* characterization);</pre>

Interface ICalibrator4

Last Modified:

15-Jun-10 Updated for 12 User chars

ECalID Property

Description	Selects the model and serial number of the ECal module to be characterized. This command does NOT set the model and serial number of the ECal module.
VB Syntax	<i>userChar.ECalID = value</i>
Variable	(Type) - Description
<i>userChar</i>	An ECalUserCharacterizer Object
<i>value</i>	(String) Model and serial number of the ECal module to be characterized.
Return Type	String
Default	"" (Empty String)
Examples	<code>userChar.ECalID = "N4433A,00001"</code>
C++ Syntax	HRESULT put_ECalID(BSTR id);
Interface	IECalUserCharacterizer

Last Modified:

2-Nov-2008 New topic (8.33)

ECALIsolation Property

Description **Note:** The inherent isolation of the PNA is better than that attained with this command. ONLY use this command when using an external test set, and ONLY using a 8509x ECal module.

Specifies whether the acquisition of the ECal calibration should include isolation or not.

VB Syntax *cal*.**ECALIsolation** = *value*

Variable [\(Type\)](#) - Description

cal A Calibrator (**object**)

value (**boolean**)

False - Exclude Isolation

True - Include Isolation

Return Type Boolean

Default False

Examples

```
Dim oPNA as AgilentPNA835x.Application
Dim oCal as Calibrator
Set oPNA = CreateObject("AgilentPNA835x.Application",
"MachineName")
Set oCal = oPNA.ActiveChannel.Calibrator
' Uncomment the following line to have the cal include isolation
' oCal.ECALIsolation = True
' Uncomment the following line to have the cal omit isolation
' oCal.ECALIsolation = False
oCal.DoECAL2Port ' Do the cal
```

C++ Syntax HRESULT put_ECALIsolation (VARIANT_BOOL bIsolationState);

HRESULT get_ECALIsolation (VARIANT_BOOL *bIsolationState);

Interface Calibrator

Last Modified:

16-Apr-2007 Un-obsolete

Read-only

ECALModuleNumberList Property

Description Returns a list of index numbers to be used for referring to the ECal modules that are currently attached to the PNA.

VB Syntax *clist* = *cal*.**ECALModuleNumberList**

Variable [\(Type\)](#) - Description

clist Variable to store the returned list of index numbers.

cal [Calibrator](#) (object)

Return Type Variant

Default Not Applicable

Examples

```
clist = cal.ECALModuleNumberList
'If 2 modules are attached to the PNA
'then the returned list will be:
1,2
```

C++ Syntax HRESULT get_ECALModuleNumberList(VARIANT *modules);

Interface ICalibrator6

EcalOrientation Property

Description Specifies which port of the ECal module is connected to which port of the PNA when the [AutoOrient](#) property = False.

VB Syntax `SMC.EcalOrientation (mod) = value`

Variable [\(Type\)](#) - Description

SMC [SMCType](#) (object)

mod (Long)

1 - Use ECAL Module for the calibration.

value **(string)** -Format this parameter in the following manner:

`Aw,Bx,Cy,Dz`

where

- A, B, C, and D are literal ports on the ECAL module
- w,x,y, and z are substituted for PNA port numbers to which the ECAL module port is connected.

Ports of the module which are not used are omitted from the string.

For example, on a 4-port ECal module with

- port A connected to PNA port 2
- port B connected to PNA port 3
- port C not connected
- port D connected to PNA port 1

the string would be: `A2,B3,D1`

Return Type String

Default "A1,B2"

Examples `SMC.EcalOrientation (1) = "A2,B1"`

C++ Syntax HRESULT put_EcalOrientation(long lModuleNum, BSTR orientation);
HRESULT get_EcalOrientation(long lModuleNum, BSTR *orientation);

Interface SMCType

EcalOrientation1Port Property

Description	For Mixer Characterization ONLY Specifies which port of the ECal module is connected to which port of the PNA for the Do1PortECAL property when the AutoOrient property = False.
VB Syntax	<code>VMC.EcalOrientation1Port (mod) = value</code>
Variable	(Type) - Description
<code>VMC</code>	VMCType (object)
<code>mod</code>	(Long)
	1 - Use ECAL Module for the calibration.
<code>value</code>	(string) - Choose from: "A1" - ECAL module port A is connected to PNA port 1 "B1" - ECAL module port A is connected to PNA port 1
Return Type	String
Default	"A1"
	If anything other than port 1 is specified, "B1" will be used. For example, if "A2" is specified, "B1" is used.
Examples	<code>VMC.EcalOrientation1Port(1) = "B1"</code>
C++ Syntax	<code>HRESULT put_EcalOrientation1Port(long lModuleNum, BSTR orientation);</code> <code>HRESULT get_EcalOrientation1Port(long lModuleNum, BSTR</code>
Interface	VMCType

EcalOrientation2Port Property

Description Specifies which port of the ECal module is connected to which port of the PNA for the [Do2PortECAL](#) property when the [AutoOrient](#) property = False.

VB Syntax `VMC.EcalOrientation2Port (mod) = value`

Variable [\(Type\)](#) - Description

VMC [VMCType](#) (**object**)

mod (Long) Module being used for the calibration.

Choose from 1 or 2.

value (**string**) -Format this parameter in the following manner:

`Aw,Bx,Cy,Dz`

where

- A, B, C, and D are literal ports on the ECAL module
- w,x,y, and z are substituted for PNA port numbers to which the ECAL module port is connected.

Ports of the module which are not used are omitted from the string.

For example, on a 4-port ECal module with

- port A connected to PNA port 2
- port B connected to PNA port 3
- port C not connected
- port D connected to PNA port 1

the string would be: `A2,B3,D1`

Return Type String

Default "A1,B2"

Examples `VMC.EcalOrientation1Port(1) = "A2,B1"`

C++ Syntax HRESULT put_EcalOrientation2Port(long lModuleNum, BSTR orientation);
HRESULT get_EcalOrientation2Port(long lModuleNum, BSTR *orientation);

Interface VMCType

ECALPortMapEx Property

Description This property replaces [ECALPortMap Property](#)

Specifies which ports of the ECal module are connected to which ports of the PNA for the [DoECAL1PortEx](#) and [DoECAL2PortEx](#) methods when the [OrientECALModule](#) property = False.

This setting remains until the PNA is restarted or this command is sent again.

Note: For guided calibrations where Orient is OFF and the same ECal module is used in more than one Connection Step, you are not allowed to specify how the ECal module is connected. Instead, the PNA determines the orientation. The PNA does not verify that you made the connection properly.

This command, and [OrientECALModule_Property](#), can be used to perform ECal orientation using the [Guided Calibration](#) interface.

VB Syntax `cal.ECALPortMapEx (module) = value`

Variable [\(Type\) - Description](#)

cal A Calibrator (**object**)

module (**long integer**) Optional argument. ECal module.

Choose from modules **1** through **8**

Use [IsECALModuleFoundEx](#) to determine the number of modules connected to the PNA

Use [GetECALModuleInfoEx](#) to return the model and serial number of each module.

value (**string**) -Format this parameter in the following manner:

Aw,Bx,Cy,Dz

where

- A, B, C, and D are literal ports on the ECAL module
- w,x,y, and z are substituted for PNA port numbers to which the ECAL module port is connected.

Ports of the module which are not used are omitted from the string.

For example, on a 4-port ECal module with

- port A connected to PNA port 2
- port B connected to PNA port 3
- port C not connected
- port D connected to PNA port 1

the string would be: A2,B3,D1

DoECAL1PortEx or **DoECAL2PortEx** methods will fail if the port numbers passed to those methods are not in the string of this property and [OrientECALModule](#) property = False.

Return Type String

Default Not Applicable

Examples

```
Dim cal As Calibrator
Dim sPortMap As String
Set cal = PNAapp.ActiveChannel.Calibrator
cal.ECALPortMapEx = "a2,b1" 'Write
sPortMap = cal.ECALPortMap 'Read
```

C++ Syntax HRESULT put_ECALPortMapEx(long moduleNumber, BSTR strPortMap);
HRESULT get_ECALPortMapEx(long moduleNumber, BSTR *strPortMap);

Interface ICalibrator4

Last Modified:

7-May-2007 Added note about orient

ElecDelayMedium Property

Description Sets or returns the electrical delay medium.

VB Syntax *meas.ElecDelayMedium = value*

Variable [\(Type\)](#) - Description

meas A Measurement (**object**)

value (**enum NACalStandardMedium**) choose from

0 - naCoax

1 - naWaveGuide

Return Type NACalStandardMedium

Default Not Applicable

Examples `Print meas.ElecDelayMedium 'prints the value of the electrical delay medium`

C++ Syntax `HRESULT get_ElecDelayMedium(tagNACalStandardMedium *pVal);`

`HRESULT put_ElecDelayMedium(tagNACalStandardMedium newVal);`

Interface IMeasurement2

ElecDistanceDelay Property

Description Sets the electrical delay in physical length (distance) for the selected measurement.

VB Syntax *meas.ElecDistanceDelay = value*

Variable [\(Type\)](#) - Description

meas A [Measurement](#) (object)

value (**double**) - Electrical Delay in distance.

Set units using [ElecDistanceDelayUnit](#).

Return Type Double

Default 0

VB `meas.ElecDistanceDelay = 1e-3 'Write`

Examples `edelay = meas.ElecDistanceDelay 'Read`

C# `Meas.ElecDistanceDelay = 1e-3 'Write`

Examples

```
// This property returns an object, and the object must be cast to a double to access the value.
```

```
Edelay = (double)meas.ElecDistanceDelay 'read
```

C++ Syntax HRESULT get_ElecDistanceDelay(VARIANT *pVal)
HRESULT put_ElecDistanceDelay(VARIANT newVal)

Interface IMeasurement11

Last Modified:

12-Nov-2010 Added C# example

6-Feb-2009 MX New topic

ElecDistanceDelayUnit Property

Description Sets and returns the units for specifying electrical delay in physical length (distance).

VB Syntax *meas.ElecDistanceDelayUnit* = *value*

Variable [\(Type\)](#) - Description

meas A [Measurement](#) (object)

value (enum naDistanceUnit) Choose from:

0 – naDistanceUnit_Meter

1 – naDistanceUnit_Feet

2 – naDistanceUnit_Inch

Return Type Enum

Default 0 – naDistanceUnit_Meter

Examples `meas.ElecDistanceDelayUnit = naDistanceUnit_Meter` **'Write**

`edelay = meas.ElecDistanceDelayUnit` **'Read**

C++ Syntax HRESULT get_ElecDistanceDelayUnit(tagNADistanceUnit *pVal)
HRESULT put_ElecDistanceDelayUnit(tagNADistanceUnit newVal)

Interface IMeasurement11

Last Modified:

6-Feb-2009 MX New topic

ElectricalDelay Property

Description Sets the Electrical Delay for the active channel.

VB Syntax *meas*.**ElectricalDelay** = *value*

Variable [\(Type\)](#) - Description

meas A Measurement (**object**)

value (**double**) - Electrical Delay in seconds. Choose any number between **-9.99** and **9.99**

Return Type Double

Default 0

Examples

```
meas.ElectricalDelay = 1e-3 'Write
edelay = meas.ElectricalDelay 'Read
```

C++ Syntax HRESULT get_ElectricalDelay(double *pVal)
HRESULT put_ElectricalDelay(double newVal)

Interface IMeasurement

Element Property

Description Returns a handle to the specified [PathElement](#) object. Each element object contains a unique set of values. The [Value Property](#) is used to set the value for each element.

This command is used to set both RF and IF Path Configuration.

- See [RF Path Configuration \(elements, value\)](#)
- See [IF Path Configuration \(elements,value\)](#)

VB Syntax Set elem = *pathConfig*.**Element** (*element*)

Variable [\(Type\)](#) - Description

elem **(Object)** [IPathElement](#)

pathConfig A [PathConfiguration](#) **(object)**

element **(String)** Configurable element. Use [pathConfig.Elements](#) to return a list of configurable elements or [see a list of configurable elements](#) for various PNA models.

Return Type Object

Default Not Applicable

Examples See examples:

[IF Path Config](#)

[RF Path Config](#)

C++ Syntax HRESULT Element(BSTR elemName, IPathElement** ppElement);

Interface IPathConfiguration

Last Modified:

27-Mar-2012 Several edits

14-Dec-2006 MX New topic

Elements Property

Description Returns an array containing the names of configurable elements.

See a [list of configurable elements and settings](#) for various PNA models.

VB Syntax *values* = *pathConfig*.**Elements**

Variable **(Type) - Description**

values **(Variant array)** Variable to store the array of configurable elements.

pathConfig A [PathConfiguration](#) **(object)**

Return Type Variant array

Default Not Applicable

Examples `elems=pathconfig.Elements`

C++ Syntax HRESULT Elements(VARIANT* pElements);

Interface IPathConfiguration

Last Modified:

14-Dec-2006 MX New topic

Embed4PortA Property

Description	Returns the PNA port number associated with 'a' based on the device topology. To see 'a' for all topologies, and to specify the port connections, use Embed4PortList Property Specify topology using Embed4PortTopology Property
VB Syntax	<i>value</i> = <i>fixture</i> . Embed4PortA
Variable	(Type) - Description <i>value</i> (Short Integer) Variable to store the returned PNA port number. <i>fixture</i> A Fixturing (object)
Return Type	Integer
Default	Not Applicable
Examples	<pre>value = fixture.Embed4PortA 'Read'</pre>
C++ Syntax	HRESULT get_Embed4PortA(short *portA);
Interface	IFixturing2

Embed4PortB Property

Description Returns the PNA port number associated with 'b' based on the device topology.

To see 'b' for all topologies, and to specify the port connections, use [Embed4PortList Property](#).

Specify topology using [Embed4PortTopology Property](#)

VB Syntax *value* = *fixture*.**Embed4PortB**

Variable [\(Type\)](#) - Description

value (Short Integer) Variable to store the returned PNA port number.

fixture A [Fixturing](#) (**object**)

Return Type Integer

Default Not Applicable

Examples `value = fixture.Embed4PortB 'Read'`

C++ Syntax HRESULT get_Embed4PortB(short *portB);

Interface IFixturing2

Embed4PortC Property

Description	Returns the PNA port number associated with 'c' based on the device topology. To see 'c' for all topologies, and to specify the port connections, use Embed4PortList Property . Specify topology using Embed4PortTopology Property
VB Syntax	<i>value</i> = <i>fixture</i> . Embed4PortC
Variable	(Type) - Description
	<i>value</i> (Short Integer) Variable to store the returned PNA port number.
	<i>fixture</i> A Fixturing (object)
Return Type	Integer
Default	Not Applicable
Examples	<pre>value = fixture.Embed4PortC 'Read'</pre>
C++ Syntax	HRESULT get_Embed4PortC(short *portC);
Interface	IFixturing2

Embed4PortD Property

Description Returns the PNA port number associated with '**d**' based on the device topology.

To see '**d**' for all topologies, and to specify the port connections, use [Embed4PortList Property](#).

Specify topology using [Embed4PortTopology Property](#)

VB Syntax *value* = *fixture*.**Embed4PortD**

Variable [\(Type\)](#) - Description

value (Short Integer) Variable to store the returned PNA port number.

fixture A [Fixturing](#) (**object**)

Return Type Integer

Default Not Applicable

Examples `value = fixture.Embed4PortD 'Read`

C++ Syntax HRESULT get_Embed4PortD(short *portD);

Interface IFixturing2

Embed4PortList Property

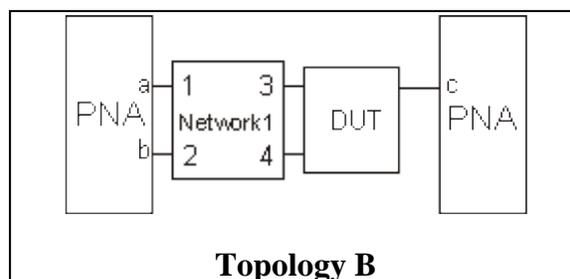
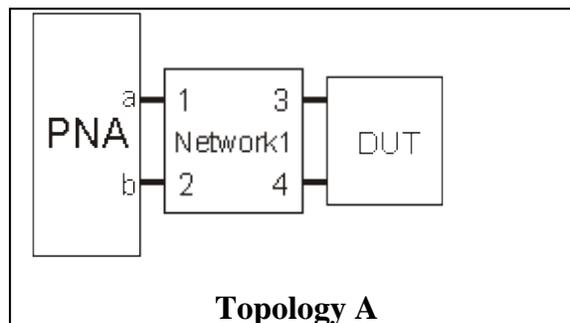
Description Specifies the PNA port connections for ALL topologies. The port assignment is dependent on the DUT topology. All four port numbers are required. However, for:

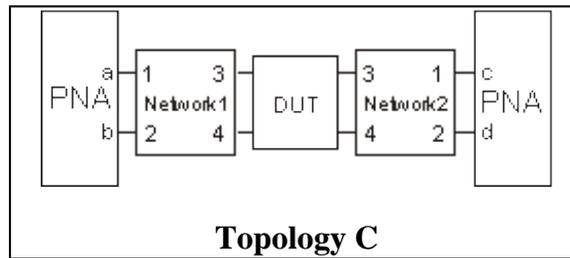
- Topology A, only the first **two** arguments are valid,
- Topology B, only the first **three** arguments are valid,
- Topology C, **ALL** arguments are valid.

Specify topology using [Embed4PortTopology Property](#).

Read the port assignments using the following commands. A, B, C, and D, refer to the port; NOT the topology.

- [Embed4PortA Property](#)
- [Embed4PortB Property](#)
- [Embed4PortC Property](#)
- [Embed4PortD Property](#)





VB Syntax `fixture.Embed4PortList = p1, p2, p3, p4`

Variable [\(Type\) - Description](#)

fixture A [Fixturing](#) (object)

p1 PNA Port number assigned to **a** in above graphic.

p2 PNA Port number assigned to **b** in above graphic.

p3 PNA Port number assigned to **c** in above graphic.

p4 PNA Port number assigned to **d** in above graphic.

Return Type Four Integers

Default 1,2,3,4

Examples `fixture.4PortNetworkTopoCPorts = 4,3,2,1` **Write**

C++ Syntax HRESULT put_4PortNetworkTopoCPorts(short ChannelNum, short pPortA, short pPortB, short pPortC, short pPortD)

Interface IFixturing2

Embed4PortNetworkFilename Property

Description Specifies the 4-port touchstone file (*.s4p) in which the network to embed or de-embed resides. If the specified file does not exist, an error occurs when type command is sent.

Following this command, send [Embed4PortNetworkMode Property](#).

Note: This command affects ALL measurements on the channel.

VB Syntax `fixture.Embed4PortNetworkFilename(netNum) = value`

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

netNum (**Integer**) Network position. Choose from **1** or **2**. See [Embed4PortTopology Property](#)

value (**String**) Full path, file name, and extension (.s4P) of the circuit.

Files are typically stored in "c:\users\public\network analyzer\documents"

Return Type String

Default Not Applicable

Examples `fixture.Embed4PortNetworkFilename(2) = "c:\users\public\network analyzer\documents/myFile.s4p" 'Write`

`value = fixture.Embed4PortNetworkFilename(1) 'Read`

C++ Syntax `HRESULT get_Embed4PortNetworkFilename(short networkNum, BSTR *filename);`

`HRESULT put_Embed4PortNetworkFilename(short networkNum, BSTR filename);`

Interface IFixturing2

Embed4PortNetworkMode Property

Description Specify the type of processing to take place on the specified 4-port network. First specify the network filename with [FSim.Embed4PortNetworkFilename Property](#).

VB Syntax `fixture.Embed4PortNetworkMode(netNum) = value`

Variable [\(Type\) - Description](#)

fixture A [Fixturing \(object\)](#)

netNum **(Integer)** Network position. Choose from **1** or **2**. See [Embed4PortTopology Property](#)

value **(Enum as NA4PortEmbedNetworkMode)** Processing mode. Choose from:

- **0** or **naNO_NETWORK** - The same as disabling.
- **1** or **naEMBED_NETWORK** - Add Network circuit.
- **2** or **naDEEMBED_NETWORK** - Remove Network circuit

Return Type Enum

Default naNO_NETWORK

Examples

```
fixture.Embed4PortNetworkMode(1) = naNO_NETWORK 'Write  
value = fixture.Embed4PortNetworkMode(2) 'Read
```

C++ Syntax

```
HRESULT get_Embed4PortNetworkMode( short networkNum,  
tagNA4PortEmbedNetworkMode *eVal );  
HRESULT put_Embed4PortNetworkMode( short networkNum,  
tagNA4PortEmbedNetworkMode eVal );
```

Interface IFixturing2

Embed4PortState Property

Description Turns ON or OFF 4-port Network embedding for all ports on the channel.

VB Syntax *fixture*.Embed4PortState = *value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

value (**Boolean**)

False - Turns Embedding OFF

True - Turns Embedding ON

Return Type Boolean

Default False (OFF)

Examples `fixture.Embed4PortState = False 'Write`

`value = fixture.Embed4PortState 'Read`

C++ Syntax HRESULT get_Embed4PortState(VARIANT_BOOL *pVal)
HRESULT put_Embed4PortState(VARIANT_BOOL newVal)

Interface IFixturing2

Embed4PortTopology Property

Description	Specifies the PNA / DUT topology. Learn more about these and other PNA/DUT configurations.
VB Syntax	<i>fixture</i> .Embed4PortTopology = <i>value</i>
Variable	(Type) - Description
<i>fixture</i>	A Fixturing (object)
<i>value</i>	(Enum as NA4PortEmbedTopology) PNA / DUT topology. Choose from: 0 or naTOPOLOGY_A - 2 PNA/DUT Ports 1 or naTOPOLOGY_B - 3 PNA/DUT Ports 2 or naTOPOLOGY_C - 4 PNA/DUT Ports
Return Type	Enum
Default	naTOPOLOGY_A (2 PNA/DUT Ports)
Examples	<pre>fixture.Embed4PortTopology = naTOPOLOGY_A 'Write value = fixture.Embed4PortTopology 'Read</pre>
C++ Syntax	<pre>HRESULT get_Embed4PortTopology(tagNA4PortEmbedTopology *eVal); HRESULT put_Embed4PortTopology(tagNA4PortEmbedTopology eVal);</pre>
Interface	IFixturing2

Enable Property

Description Turns ON / OFF the trigger output.

VB Syntax *auxTrig.Enable* = *state*

Variable [\(Type\)](#) - Description

auxTrig An [AuxTrigger](#) (object)

state (boolean)

True - Trigger Output ON

False - Trigger Output OFF

Return Type Boolean

Default False

Exaamples `auxTrig.Enable = True 'Write`

`value = auxTrig.Enable 'Read`

C++ Syntax HRESULT get_Enable(VARIANT_BOOL * enable);

HRESULT put_Enable(VARIANT_BOOL enable);

Interface IAuxTrigger

Last Modified:

14-Dec-2006 MX New topic

EnableAllOutput Property

Description Sets and returns the ON / OFF state of all DC sources for the channel.

VB Syntax `dc.EnableAllOutput = state`

Variable [\(Type\)](#) - Description

dc An [DCStimulus](#) (object)

state (boolean)

True - All DC sources ON

False - All DC sources OFF

Return Type Boolean

Default False

Examples `dc.EnableAllOutput = True 'Write`

`value = dc.EnableAllOutput 'Read`

C++ Syntax HRESULT get_EnableAllOutput(VARIANT_BOOL * pValue);
HRESULT put_EnableAllOutput(VARIANT_BOOL newValue);

Interface IDCStimulus

Last Modified:

21-Feb-2012 MX New topic

Enabled Property

Description Enables and disables (ON/OFF) the port mapping and control line output of the specified test set.

If the specified test set is not connected or not ON, then setting Enabled = True will report an error. All other properties can be set when the test set is not connected.

When this command is set to ON or OFF, then the display of the test set status bar ([ShowProperties Property](#)) is also set to ON or OFF.

VB Syntax *tset.Enabled* = *value*

Variable [\(Type\)](#) - Description

tset A [TestsetControl](#) object

OR

An [E5091Testset](#) object

value **(Boolean)**

True Enables test set control.

False Disables test set control.

Return Type Boolean

Default False

Examples [See E5091A Example Program](#)

[See External Testset Program](#)

C++ Syntax HRESULT get_Enabled(VARIANT_BOOL *state);
HRESULT put_Enabled(VARIANT_BOOL state);

Interface ITestsetControl

IE5091Testsets

EnableLOPowerCal Property

Description Sets and returns whether or not the LO power cal step is included in the cal steps when a Cal is performed.

VB Syntax *obj.EnableLOPowerCal(n) = value*

Variable [\(Type\)](#) - Description

obj A [SMCType](#) (object)

A [VMCType](#) (object)

A [SweptIMDCal](#) (object)

A [NoiseCal](#) (object)

n LO Stage. Choose 1. (Only single LO allowed)

value (Boolean) Choose from:

False - Skips over the LO Power Cal when calibrating.

True - Includes a step for LO Power Cal when calibrating.

Return Type Boolean

Default PNA Rev. 9.1 and above: **False**

Before Rev 9.1: **True**

Examples

```
imd.EnableLOPowerCal(1)= true 'Write
```

```
loPwrCal = imd.EnableLOPowerCal(1) 'Read
```

C++ Syntax HRESULT get_EnableLOPowerCal (long stage, BOOL *enable)

HRESULT put_EnableLOPowerCal (long stage, BOOL enable)

Interface ISMCType

IVMCType

ISweptIMD2

INoiseCal2

6-Oct-2011 Added SMC and VMC
21-Oct-2009 Added NoiseCal
27-Mar-2009 MX New topic

EnablePhase Property

Description Sets and returns the state of SMC Phase measurements.

In the User Interface, there are two "enable phase" checkboxes: in the [Phase Settings dialog](#) and in the [Calibration Wizard](#). Checking one enables both. This single command also enables both.

VB Syntax *obj.EnablePhase = bool*

Variable [\(Type\)](#) - Description

obj A [Mixer Interface](#) pointer to the [Measurement](#) (object)

A [Converter](#) (Object)

bool **(Boolean)** -

True - Include phase in SMC measurements

False - Do NOT include phase in SMC measurements

Return Type Boolean

Default False

Examples `mixer.EnablePhase = True`

C++ Syntax `HRESULT get_EnablePhase(VARIANT_BOOL * val);`

`HRESULT put_EnablePhase(VARIANT_BOOL val);`

Interface IMixer13

Last Modified:

25-Mar-2010 MX New topic

EnablePowerCompensation Property

Description Adjusts the source power at the specified port by the combined amount of loss through ALL enabled fixturing operations. Use this function to set the power level at the DUT input. [Learn more.](#)

Note: This command affects ALL measurements on the specified channel.

VB Syntax `fixture.EnablePowerCompensation (port) = bool`

Variable [\(Type\) - Description](#)

fixture A [Fixturing \(object\)](#)

port (**Integer**) Port number to receive power compensation.

bool (Boolean)

True - Compensate source power

False - Do NOT compensate source power

Return Type Boolean

Default False

Examples

```
fixture.EnablePowerCompensation(1) = True
value = fixture.EnablePowerCompensation(2) 'Read
```

C++ Syntax `HRESULT get_EnablePowerCompensation(short port, VARIANT_BOOL *pState);`
`HRESULT put_EnablePowerCompensation(short port, VARIANT_BOOL bVal);`

Interface IFixturing5

Last Modified:

13-Apr-2010 MX New topic

EnableSnPDataExtrapolation

Description Turns ON and OFF SNP file extrapolation for both 2-port and 4-port embedding/de-embedding.

VB Syntax `fixture.EnableSnPDataExtrapolation = bool`

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

bool **True** - Turns Extrapolation ON

False - Turns Extrapolation OFF

Return Type Boolean

Default False

Examples

```
fixture.EnableSnPDataExtrapolation = True  
value = fixture.EnableSnPDataExtrapolation 'Read
```

C++ Syntax HRESULT get_EnableSnPDataExtrapolation(VARIANT_BOOL *pExtrap);
HRESULT put_EnableSnPDataExtrapolation(VARIANT_BOOL bExtrap);

Interface IFixturing6

Last Modified:

16-Nov-2010 MX New topic

EnableSourceUnleveledEvents Property

Description Specifies whether or not to report [Source Unleveled](#) errors as system events. These events can trigger an [OnSystemEvent](#) call.

This setting will revert to the default (False) setting on Instrument Preset.

VB Syntax `pref.EnableSourceUnleveledEvents = bool`

Variable [\(Type\)](#) - Description

pref A [Preferences](#) (object)

bool (**Boolean**) - Choose from:

False - Do NOT report Source Unleveled Errors.

True - Report Source Unleveled Errors.

Return Type Boolean

Default False

Examples

```
pref.EnableSourceUnleveledEvents = False 'Write
```

```
prefer = pref.EnableSourceUnleveledEvents 'Read
```

C++ Syntax

```
HRESULT put_EnableSourceUnleveledEvents( VARIANT_BOOL bsourcUnlEnable)
```

```
HRESULT get_EnableSourceUnleveledEvents( VARIANT_BOOL *bsourcUnlEnable)
```

Interface IPreferences3

Last modified:

9-Apr-2010 Added Preset note
15-Nov-2006 MX New command

EndOfSweepOperation Property

Description Set and read the action which should be taken at the end of the last frequency or power sweep in the measurement. This setting is used to protect a sensitive device from too much power during the sweep retrace.

VB Syntax `gca.EndOfSweepOperation = value`

Variable [\(Type\)](#) - Description

gca A [GainCompression](#) (object)

value (**NAGCAEndOfSweepOperation**)

- **naStandard (0)** Use the default PNA method. [Learn more.](#)
- **naSetToStartPower (1)** Sweep Start power
- **naSetToStopPower (2)** Sweep Stop power.
- **naSetRFOff (3)** Always turn power OFF while waiting.

Return Type Enum

Default naStandard

Examples

```
gca.EndOfSweepOperation = naSetToStartPower 'Write
eos = gca.EndOfSweepOperation 'Read
```

C++ Syntax HRESULT get_EndOfSweepOperation(tagNAGCAEndOfSweepOperation* pVal)
 HRESULT put_EndOfSweepOperation(tagNAGCAEndOfSweepOperation newVal)

Interface IGainCompression

Last Modified:

11-Sep-2007 MX New topic

ENRFile Property

Description Sets and returns the name of the ENR file associated with the noise source.

VB Syntax *noise*.**ENRFile** = *value*

Variable [\(Type\)](#) - Description

noise A [NoiseCal](#) (**object**)

value (**string**) Full path and ENR filename.

Return Type String

Default Not Applicable

Examples

```
noise.ENRFile = "c:/ProgramFiles/Agilent/Network  
Analyzer/Documents/ENR/346C.enr" 'Write
```

```
ENR = noise.ENRFile 'Read
```

C++ Syntax HRESULT get_ENRFile(BSTR* pValue)

HRESULT put_ENRFile(BSTR pNewValue)

Interface INoiseCal

Last Modified:

29-May-2007 MN New topic

ENRID Property

Description Sets and returns ID of ENR table.

VB Syntax *enr*.ENRSN = *ID*

Variable [\(Type\)](#) - Description

enr An [ENRFile](#) (**object**)

ID Identifier for the ENR table

Return Type String

Default Not Applicable

Examples [See example program](#)

C++ Syntax HRESULT get_ENRID(BSTR *Val);
HRESULT put_ENRID(BSTR Val);

Interface IENRFile

Last Modified:

2-Aug-2007 MX New topic

ENRSN Property

Description	Sets and returns the serial number of the noise source for which the ENR file applies.
VB Syntax	<i>enr.ENRSN = serialNumber</i>
Variable	(Type) - Description
<i>enr</i>	An ENRFile (object)
<i>serialNumber</i>	(String) Serial number of the noise source.
Return Type	String
Default	Not Applicable
Examples	See example program
C++ Syntax	<pre>HRESULT get_ENRSNBSTR *Val); HRESULT put_ENRSN(BSTR Val);</pre>
Interface	IENRFile

Last Modified:

2-Aug-2007 MX New topic

EqualTonePower Property - Superseded

Description	Note: This command is replaced with LevelingMethod Property Sets and returns the ON OFF state of Equal Tone Power for the Swept IMD or IMSpectrum measurement.
VB Syntax	<i>object</i> . EqualTonePower = <i>value</i>
Variable	(Type) - Description
<i>object</i>	A SweptIMD or IMSpectrum Object
<i>value</i>	(Boolean) - Choose from: True - Set equal tone power. False - Do NOT set equal tone power.
Return Type	Boolean
Default	False
Examples	<pre>ims.EqualTonePower = true 'Write value = imd.EqualTonePower 'Read</pre>
C++ Syntax	HRESULT get_EqualTonePower(VARIANT_BOOL* val) HRESULT put_EqualTonePower(VARIANT_BOOL val)
Interface	ISweptIMD2 IIMSpectrum2

Last modified:

14-Nov-2012 Superseded

5-May-2011 New topic

ErrorCorrection Property

Description Sets (or returns) error correction ON or OFF for the measurement.

VB Syntax *meas*.**ErrorCorrection** = *value*

Variable [\(Type\)](#) - Description

meas A Measurement (**object**)

value (**boolean**)

False - Turns error correction OFF

True - Turns error correction ON

Return Type Boolean

Default See [Error Correction](#)

Examples `meas.ErrorCorrection = True 'Write`

`errcorr = meas.ErrorCorrection 'Read`

C++ Syntax HRESULT put_ErrorCorrection (VARIANT_BOOL bState)
HRESULT get_ErrorCorrection (VARIANT_BOOL *bState)

Interface IMeasurement

ErrorCorrection (Channel) Property

Description Attempts to sets error correction ON or OFF for all of the measurements on the channel. This setting may not be successful for some measurements because the Cal Set currently in place may not contain the appropriate calibration data. To read the error correction state for a measurement, use [Error Correction Property](#).

VB Syntax *chan*.**ErrorCorrection** = *value*

Variable [\(Type\)](#) - Description

chan A [Channel](#) (**object**)

value (**boolean**)

False - Turns error correction OFF

True - Turns error correction ON

Return Type Boolean

Default [About Error Correction](#)

Examples `chan.ErrorCorrection = True`

C++ Syntax HRESULT put_ErrorCorrection (VARIANT_BOOL bState)

Interface IChannel7

ErrorCorrectionIndicator Property

Description	Returns the error correction state for the measurement.
VB Syntax	<i>value</i> = <i>meas</i> . ErrorCorrectionIndicator
Variable	(Type) - Description
<i>value</i>	(Enum) Error correction state. <ul style="list-style-type: none">• 0 - naErrorCorrectionIndicator_None - No error correction• 1 - naErrorCorrectionIndicator_Master - Original error correction terms applied.• 2 - naErrorCorrectionIndicator_Interpolated - Error terms are interpolated. Learn more• 3 - naErrorCorrectionIndicator_Delta - Delta Match calibration terms. Learn more• 4 - naErrorCorrectionIndicator_Invalid Error terms are not valid.
<i>meas</i>	A Measurement (object)
Return Type	Enum as NAErrorCorrectionIndicator
Default	See Error Correction
Examples	<pre>errcorr = meas.ErrorCorrectionIndicator 'Read</pre>
C++ Syntax	HRESULT get_ErrorCorrectionIndicator (enum NAErrorCorrectionIndicator *pIndicator);
Interface	IMeasurement14

Last Modified:

14-Jul-2010 MX New topic

Write-only

ExtendedProperties Property

Description Provides access to the custom properties and methods of an external device.

VB Syntax Set *PSG* = *ExtDev*.**ExtendedProperties**

Variable [\(Type\)](#) - Description

PSG **(Object)** Variable to store the returned handle to an external device.

ExtDev An [ExternalDevice](#) **(object)**

Return Type Object

Default Not Applicable

Examples [See Example](#)

C++ Syntax HRESULT get_ExtendedProperties(IDispatch** ppObject);

Interface IExternalDevice

Last Modified:

27-Aug-2009 MX New topic

Write/Read

ExternalALC Property

Description Sets or returns the source of the analyzer leveling control.

VB Syntax *app.ExternalALC = value*

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

value (**boolean**) - Choose from:

True - Leveling control supplied through the rear panel.

False - Leveling control supplied inside the analyzer

Return Type Boolean

Default False

Examples `app.ExternalALC = True 'Write`

`extALC = app.ExternalALC 'Read`

C++ Syntax HRESULT get_ExternalALC(VARIANT_BOOL *pVal)
HRESULT put_ExternalALC(VARIANT_BOOL newVal)

Interface IApplication

ExternalDeviceDeActivatePolicy Property

Description Set and return whether External Devices remain activated or are de-activated when the PNA is Preset or when a Instrument State is recalled.

This setting remains until changed again using this command, or until the hard drive is changed or reformatted.

See the [ExternalDevices](#) collection.

VB Syntax `pref.ExternalDeviceDeActivatePolicy = value`

Variable [\(Type\)](#) - Description

pref A [Preferences](#) (object)

value (**Boolean**) - Choose from:

- **True** - External device are **de-activated** when the PNA is Preset or when a Instrument State is recalled.
- **False** - External devices **remain active** when the PNA is Preset or when a Instrument State is recalled.

Return Type Boolean

Default **True**

Examples

```
pref.ExternalDeviceDeActivatePolicy = 1 'Write
```

```
dDevPolicy = pref.ExternalDeviceDeActivatePolicy 'Read
```

C++ Syntax HRESULT get_ExternalDeviceDeActivatePolicy(VARIANT_BOOL * pref);

HRESULT put_ExternalDeviceDeActivatePolicy(VARIANT_BOOL pref);

Interface IPreferences10

Last Modified:

17-Sep-2009 MX New topic

ExternalTriggerConnectionBehavior Property

Description Configures the external triggering signal for the PNA and PNA-X.

- [TriggerSouce Property](#) is automatically set to External when **ExternalTriggerConnectionBehavior** is sent.
- Edge triggering is only available on some PNA models.
- For more information, see [External Triggering](#).

VB Syntax *trigsetup*.**ExternalTriggerConnectionBehavior** (*conn*) = *value*

Variable [\(Type\)](#) - Description

trigsetup A [TriggerSetup](#) (object)

conn (**enum NATriggerConnection**) Rear Panel connector to send or receive trigger signals. Choose from:

Only one of the input connectors is active at a time. When a command is sent to one, the PNA automatically makes the other INACTIVE.

0 - naTriggerConnectionAUXT Trigger IN from rear-panel AUX IO connector Pin 19 (No longer supported).

1 - naTriggerConnectionBNC1 Trigger IN

- [MEAS TRIG IN](#) on models

2 - naTriggerConnectionBNC2 Trigger OUT. Only useful in point sweep mode.

- [AUX TRIG 1 OUT](#) on models

3 - naTriggerConnectionMATH Trigger IN from rear-panel [Material Handler connector Pin 18](#)

value (**enum NAExternalTriggerBehavior**) -

0 - naTriggerInactive - Disables the specified connector.

Choose from 1 through 4 when <conn> is set to naTriggerConnection**BNC1**

1 - naTriggerInEdgeNegative - Triggers the PNA when receiving a negative going

signal

2 - **naTriggerInEdgePositive** - Triggers the PNA when receiving a positive going signal

3 - **naTriggerInLevelLow** - Triggers the PNA when receiving a low level signal

4 - **naTriggerInLevelHigh** - Triggers the PNA when receiving a High-level signal

Choose from 5 through 8 when <conn> is set to **naTriggerConnectionBNC2**.

In addition to sending this command, you must also use [TriggerOutputEnabled Property](#) to enable the BNC2 output.

5 - **naTriggerOutPulsePositiveAfter** - Sends a POSITIVE going TTL pulse at the END of each point during the sweep.

6 - **naTriggerOutPulsePositiveBefore** - Sends a POSITIVE going TTL pulse at the START of each point during the sweep.

7 - **naTriggerOutPulseNegativeAfter** - Sends a NEGATIVE going TTL pulse at the END of each point during the sweep.

8 - **naTriggerOutPulseNegativeBefore** - Sends a NEGATIVE going TTL pulse at the START of each point during the sweep.

Return Type Enum as NAExternalTriggerBehavior

Default BNC1 = **naTriggerInactive**

BNC2 = **naTriggerInactive**

When [Output is enabled](#)

BNC1 = **naTriggerInactive**

BNC2 = **naTriggerOutPulsePositiveAfter**

Examples

```
trigsetup.ExternalTriggerConnectionBehavior (naTriggerConnectionBNC1) =  
naTriggerInLevelLow 'Write
```

```
trigBehav = trigsetup.ExternalTriggerConnectionBehavior  
(naTriggerConnectionAUXT) 'Read
```

C++ Syntax HRESULT get_ExternalTriggerConnectionBehavior(tagNATriggerConnection connection,tagNAExternalTriggerBehavior *trigger);

HRESULT put_ExternalTriggerConnectionBehavior(tagNATriggerConnection connection,tagNAExternalTriggerBehavior trigger);

Interface ITriggerSetup

Last Modified:

12-Apr-2012 Removed reference to old models

12-Dec-2011 Added notes for models

25-Feb-2008 Added 'Global' note

ExternalTriggerDelay Property

Description Sets and reads the trigger delay for all measurements in the CHANNEL. This delay is only applied while in [app.Source = naTriggerSourceExternal](#) and [trigsetup.Scope = naChannelTrigger](#) . After an external trigger is applied, the start of the sweep is delayed for the specified delay value plus any inherent latency.

To apply a trigger delay for all channels (Global), use [TriggerDelay Property](#).

VB Syntax *chan.ExternalTriggerDelay = value*

Variable [\(Type\)](#) - Description

chan A [Channel](#) (object)

value **Double**- Trigger delay value in seconds. Range is from 0 to 107 seconds

Return Type Double

Default 0

Examples

```
chan.ExternalTriggerDelay = .003 'Write
```

```
delay = chan.ExternalTriggerDelay 'Read
```

C++ Syntax HRESULT get_ExternalTriggerDelay(double *delay);

HRESULT put_ExternalTriggerDelay(double delay)

Interface IChannel6

F1Frequency Property

Description Sets and returns the frequency of the F1 tone. Use with IMD sweep types:

- `naIMDToneCWSweep`
- `naIMDTonePowerSweep`

VB Syntax *object.F1Frequency = value*

Variable [\(Type\)](#) - Description

object A [SweptIMD](#) or [IMSpectrum](#) Object

value (Double) F1 tone frequency in Hz. The F1 and F2 tones MUST be within the frequency range of the PNA.

Return Type Double

Default .9995 GHz

Examples `imd.F1Frequency = 100e6 'Write`

`value = imd.F1Frequency 'Read`

C++ Syntax `HRESULT get_F1Frequency(double *pVal)`
`HRESULT put_F1Frequency(double newVal)`

Interface `ISweptIMD`
`IIMSpectrum`

Last Modified:

19-Aug-2008 MX New topic

F2Frequency Property

Description Sets and returns the frequency of the F2 tone. Use with IMD sweep types:

- `naIMDToneCWSweep`
- `naIMDTonePowerSweep`

VB Syntax *object.F2Frequency = value*

Variable [\(Type\)](#) - Description

object A [SweptIMD](#) or [IMSpectrum](#) Object

value (Double) F2 tone frequency in Hz. The F1 and F2 tones MUST be within the frequency range of the PNA.

Return Type Double

Default 1.0005 GHz

Examples

```
imd.F2Frequency = 200e9 'Write
```

```
value = imd.F2Frequency 'Read
```

C++ Syntax HRESULT get_F2Frequency(double *pVal)
HRESULT put_F2Frequency(double newVal)

Interface ISweptIMD
IIMSpectrum

Last Modified:

19-Aug-2008 MX New topic

FailedTraces Property

Description Set and return the limit line color of failed traces or failure indicators (dots) and the word **Fail**.

VB Syntax *colors.FailedTraces = value*

Variable [\(Type\)](#) - Description

colors A [ComColors](#) (object)

value **(Long Integer)** - RGB color of the FailedTraces pen.

Convert the three RGB colors to an integer as follows:

```
RGB = R+(G*2^8)+(B*2^16)
```

To find the three RGB values from the [Display Colors dialog](#), click **Change Color**, then **Define Custom Color**.

Return Type Long

Default Display = 255,20,20

Print = 255,20,20

Examples

```
R = 10
G = 10
B = 10
RGB = R+(G*2^8)+(B*2^16)
colors.FailedTraces = RGB 'Write
color = colors.FailedTraces 'Read
```

C++ Syntax HRESULT get_FailedTraces(long* pVal);

HRESULT put_FailedTraces(long newVal);

Interface IComColors

Last Modified:

7-Aug-2009 MX New topic

FastCWPointCount Property

Description Enables Fast CW sweep and sets the number of data points for the channel. [Sweep Type](#) must already be set to CWTime and FIFO must already be enabled.

See Also

[FIFO and other Antenna Features](#)

[FIFO Object](#)

[Example program](#)

[N5264A Measurement Receiver](#)

VB Syntax *chan*.FastCWPointCount = *value*

Variable (Type) - Description

chan A [Channel](#) Object

value (Long Integer) Number of data points to measure in Fast CW mode. This setting overwrites the standard number of points setting for the channel.

Set to 0 to disable Fast CW.

Return Type Long Integer

Default 0

Examples

```
chan.FastCWPointCount = 1e3 'Write
```

```
value = chan.FastCWPointCount 'Read
```

C++ Syntax HRESULT get_FastCWPointCount(long *value)

HRESULT put_FastCWPointCount(long value)

Interface IChannel16

Last Modified:

2-Feb-2011 Fixed syntax

7-Oct-2008 MX New topic

FastMode Property

Description	Sets and returns the state of a separate IFBW setting for leveling sweeps. ON allows a higher (faster) IFBW than the measurement sweep. It also causes leveling sweeps to be noisier.
VB Syntax	<i>RxLevel</i> . FastMode (<i>srcPort</i>) = <i>value</i>
Variable	(Type) - Description
<i>RxLevel</i>	A ReceiverLeveling Object
<i>srcPort</i>	(Long Integer) Source port for which to set Fast Mode for Receiver Leveling.
	Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use chan.getPortNumber to translate the string into a port number. To learn more see Remotely Specifying a Source Port
<i>value</i>	(Boolean) Separate IFBW setting state. Choose from: True - Use separate IFBW setting. Specify IFBW using LevelingIFBW . False - Use same IFBW as the measurement sweep. Specify IFBW using IF Bandwidth .
Return Type	Variant Boolean
Default	True
Examples	<pre>rxLevel.FastMode (1) = True ' Write value = rxLevel.FastMode 2 ' Read</pre>
C++ Syntax	HRESULT get_FastMode(long port, VARIANT_BOOL* pVal); HRESULT put_FastMode(long port, VARIANT_BOOL newVal);
Interface	IRceiverLevelingConfiguration

Last Modified:

13-Feb-2009 MX New topic

FilterBW Property

Description Returns the results of the SearchBandwidth method.

VB Syntax `filtBW = meas.FilterBW`

Variable [\(Type\)](#) - [Description](#)

filtBW (single) - Variable to store bandwidth data

meas A Measurement (**object**)

Return Type Single

Default Not applicable

Examples `filterBW = meas.FilterBW 'Read`

C++ Syntax HRESULT get_FilterBW(float* bw)

Interface IMeasurement

FilterCF Property

Description Returns the Center Frequency result of the SearchBandwidth method.

VB Syntax `filtCF = meas.FilterCF`

Variable [\(Type\)](#) - Description

`filtCF` (**double**) - Variable to store bandwidth CF data

`meas` A Measurement (**object**)

Return Type Double

Default Not applicable

Examples `filtCF = meas.FilterCF 'Read`

C++ Syntax HRESULT get_FilterCF(double* centerFrequency)

Interface IMeasurement

FilterLoss Property

Description Returns the Loss value of the SearchBandwidth method.

VB Syntax *filtLoss* = *meas*.FilterLoss

Variable [\(Type\)](#) - Description

filtLoss (**single**) - Variable to store bandwidth Loss data

meas A Measurement (**object**)

Return Type Single

Default Not applicable

Examples `filterLoss = meas.FilterLoss 'Read`

C++ Syntax HRESULT get_FilterLoss(float* loss)

Interface IMeasurement

FilterQ Property

Description Returns the Q (quality factor) result of the SearchBandwidth method.

VB Syntax `filtQ = meas.FilterQ`

Variable [\(Type\)](#) - Description

`filtQ` (single) - Variable to store bandwidth Q data

`meas` A Measurement (**object**)

Return Type Single

Default Not applicable

Examples `filtQ = meas.FilterQ 'Read`

C++ Syntax HRESULT get_FilterQ(float* quality)

Interface IMeasurement

FilterErrors Property

Description Returns the error string associated with the digital filters. The return string has three fields separated by commas: "stage1 status, stage2 status, stage3 status"

Each of these fields can contain one or more of the following error codes:

- **NO ERROR**
- ***NUMBER-OF-COEFFICIENTS** - the number of coefficients is excessive for that filter-stage
- ***COEFFICIENT VALUE** - one or more coefficients are out of range for that filter-stage
- ***SUM-OF-COEFFICIENTS** - the sum of all coefficients is excessive for that filter-stage,
- ***FREQUENCY** - the frequency for Stage 1 is out of range (only applies stage1 field),
- ***PARAMETER** - one or more parameters are out of range (only applies to stage 3 field)

VB Syntax *value* = *spm4*.FilterErrors

Variable [\(Type\)](#) - Description

value Variable to store the returned errors.

spm4 A [SignalProcessingModuleFour](#) (object)

Return Type String

Default Not Applicable

Examples

```
mode = spm4.FilterErrors 'Read

'example return strings"

NO ERROR, NO ERROR, NO ERROR
indicates no errors,

*SUM-OF-COEFFICIENTS, NO ERROR, NO ERROR
indicates that the sum of all filter coefficients exceed the
maximum value for the Stage-1 filter,

*COEFFICIENT *SUM-OF-COEFFICIENTS, NO ERROR, *PARAMETER
indicates a problems with Stage 1 coefficients and a problem
with one or more of the parameters associated with the Stage 3
filter.
```

C++ Syntax HRESULT get_FilterErrors(BSTR* dspErrors);

Interface ISignalProcessingModuleFour

Last Modified:

1-Jan-2007 MX New topic

FilterMode Property

Description Sets and returns whether the PNA configures the 3-stage digital filter settings or they will be configured manually. When making manual settings, also send [ADCCaptureMode Property](#) which routes the IF through the 3-stage filter.

VB Syntax `spm4.FilterMode = value`

Variable (Type) - Description

spm4 A [SignalProcessingModuleFour \(object\)](#)

value (**enum as NAModes**) Filter mode. Choose from:

naAUTO PNA controls digital filter settings.

naMANUAL You control digital filter settings using other [SignalProcessingModuleFour](#) commands.

Return Type Enum

Default naAUTO

Examples

```
spm4.FilterMode = naAUTO 'Write
mode = spm4.FilterMode 'Read
```

C++ Syntax HRESULT get_FilterMode(tagNAModes* dspMode);
HRESULT put_FilterMode(tagNAModes dspMode);

Interface ISignalProcessingModuleFour

Last Modified:

24-Jan-2007 MX New topic

FirmwareMajorRevision Property

Description	Returns the major firmware revision number as an integer. For example, given a firmware revision number A.03.30, this command returns 3.
VB Syntax	<i>value</i> = <i>cap</i> . FirmwareMajorRevision
Variable	(Type) - Description
<i>value</i>	(Long) - Variable to store the returned integer value of the firmware revision number.
<i>cap</i>	A Capabilities (object)
Return Type	Long
Default	Not Applicable
Examples	<code>value = cap.FirmwareMajorRevision 'Read</code>
C++ Syntax	HRESULT get_FirmwareMajorRevision(long * majorRev);
Interface	ICapabilities

FirmwareMinorRevision Property

Description	Returns the minor firmware revision number as an integer. For example, given a firmware revision number A.03.30, this command returns 30.
VB Syntax	<i>value</i> = <i>cap</i> . FirmwareMinorRevision
Variable	(Type) - Description
<i>value</i>	(Long) - Variable to store the returned decimal value of the firmware revision number.
<i>cap</i>	A Capabilities (object)
Return Type	Long
Default	Not Applicable
Examples	<code>value = cap.FirmwareMinorRevision 'Read</code>
C++ Syntax	HRESULT get_FirmwareMinorRevision(long * minorRev);
Interface	ICapabilities

FirmwareSeries Property

Description	Returns the alpha portion of the firmware revision number. For example, given a firmware revision number A.03.30, this command returns A.
VB Syntax	<i>value</i> = <i>cap</i> . FirmwareSeries
Variable	(Type) - Description
<i>value</i>	(String) - Variable to store the returned alpha value of the firmware revision number.
<i>cap</i>	A Capabilities (object)
Return Type	String
Default	Not Applicable
Examples	<pre>value = cap.FirmwareSeries 'Read</pre>
C++ Syntax	HRESULT get_FirmwareSeries(BSTR * series);
Interface	ICapabilities

FixedDelay Property

Description Set and return the known delay through the calibration mixer.

VB Syntax *smc*.**FixedDelay** = *value*

Variable [\(Type\)](#) - Description

smc An [SMCType](#) (object)

value (Double) Known delay through the calibration mixer in seconds.

Return Type Double

Default 0 seconds

Example `SMC.FixedDelay = 12e-9` **Write**

`value =SMC.FixedDelay` **Read**

C++ Syntax HRESULT put_FixedDelay(Double Value);

HRESULT get_FixedDelay(Double* Value);

Interface SMCType5

Last Modified:

25-Mar-2010 MX New topic

FixedPhase Property

Description Write and read the fixed phase value. Must not be logarithmic sweep type.

VB Syntax *phase*.FixedPhase(*srcPort*) = *value*

Variable [\(Type\)](#) - Description

phase A [PhaseControl](#) Object

srcPort (Long Integer) Source port for which to make phase control settings.

Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#)

value (Double) Phase value in degrees. Choose a value between -360 and 360.

Return Type Double

Default 0 degrees

Examples

```
phase.FixedPhase 1 = 15 ' Write  
value = phase.FixedPhase 2 ' Read
```

C++ Syntax

```
HRESULT get_FixedPhase(long port, double* pVal);  
  
HRESULT put_FixedPhase(long port, double newVal);
```

Interface IPhaseControl

Last Modified:

8-Dec-2010 MX New topic

FixedRatioedPower Property

Description	Write and read the fixed power ratioed value. Must NOT be in power sweep to use this value during phase control.
VB Syntax	<i>phase</i> .FixedRatioedPower(<i>srcPort</i>) = <i>value</i>
Variable	(Type) - Description
<i>phase</i>	A PhaseControl Object
<i>srcPort</i>	(Long Integer) Source port for which to make phase control settings.
	Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use chan.getPortNumber to translate the string into a port number. To learn more see Remotely Specifying a Source Port
<i>value</i>	(Double) Fixed power ratio value in dBc within the allowable range of the PNA.
Return Type	Double
Default	0 dBc
Examples	<pre>phase.FixedRatioedPower 1 = -1 ' Write value = phase.FixedRatioedPower 2 ' Read</pre>
C++ Syntax	<pre>HRESULT get_FixedRatioedPower(long port, double* pVal); HRESULT put_FixedRatioedPower(long port, double newVal);</pre>
Interface	IPhaseControl

Last Modified:

8-Dec-2010 MX New topic

FixturingState Property

Description Turns all three fixturing functions (de-embedding, port matching, impedance conversion) ON or OFF for all ports on the specified channel. This does NOT affect port extensions.

VB Syntax *fixture.FixturingState* = *value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

value (boolean)

True - Turns Fixturing ON

False - Turns Fixturing OFF

Return Type Boolean

Default False

Examples `fixture.FixturingState = True 'Write`

`value = fixture.FixturingState 'Read`

C++ Syntax HRESULT get_FixturingState(VARIANT_BOOL *pVal)

HRESULT put_FixturingState(VARIANT_BOOL newVal)

Interface IFixturing

Read-only

FootSwitch Property

Description Reads the Footswitch Input (pin 20 of the AUX IO connector).

VB Syntax *value* = *AuxIO*.**Footswitch**

Variable [\(Type\)](#) - **Description**

value **(boolean)** - Variable to store the returned value

False - foot switch is released

True - footswitch is depressed

AuxIO **(object)** - A Hardware Aux I/O object

Return Type Boolean

Default True

Examples `fs = aux.Footswitch`

C++ Syntax HRESULT get_FootSwitch (VARIANT_BOOL* State);

Interface IHWAuxIO

FootswitchMode Property **Obsolete**

Description	Determines what occurs when the footswitch is pressed. For more information see the FootSwitch In pin description in the Auxiliary IO connector.
VB Syntax	<i>AuxIo.FootSwitchMode = value</i>
Variable	(Type) - Description
<i>value</i>	(enum NAFootSwitchMode)
	0 - naIgnoreFootswitch - Footswitch presses are ignored.
	1 - naSweepTrigger - Footswitch presses trigger a sweep. The PNA must be in Manual Trigger Mode.
	2 - naRecallNextState - Footswitch presses recall an instrument state. When more than one state is available, then each footswitch press recalls the next state, then starts over from the beginning. It is possible for a recalled state to override the current mode. If the recalled state is IGNore, then mode changes and additional footswitch presses are ignored.
	3 - naRunMacro - Footswitch presses load and run a macro. When more than one macro is available, then each footswitch press loads and runs the next macro, then starts over from the beginning. It is possible for a Macro to override the current mode. If the macro contains a Preset, then the mode changes to the default setting IGNore and additional footswitch presses are ignored.
<i>AuxIO</i>	(object) - A Hardware Aux I/O object
Return Type	NAFootSwitchMode
Default	0 - naIgnoreFootswitch
Examples	<code>auxIo.FootSwitchMode = naIgnoreFootSwitch 'Write</code>
C++ Syntax	HRESULT get_FootSwitchMode(NAFootSwitchMode *pFootSwitchMode) HRESULT put_FootSwitchMode(NAFootSwitchMode newFootSwitchMode)
Interface	IHWAuxIO3

ForceDeEmbedENRAdapter Property

Description Set and read the De-embedENRAdapter state. [Learn more](#).

VB Syntax `noiseCal.ForceDeEmbedENRAdapter = value`

Variable [\(Type\)](#) - Description

noiseCal A [NoiseCal](#) (object)

value (**boolean**) - ENR Adapter de-embed state.

False - Do not Force de-embedding.

True - Force de-embedding.

Return Type Boolean

Default False

Examples `noiseCal.ForceDeEmbedENRAdapter = False` **Write**
`AdapterDembed = noiseCal.ForceDeEmbedENRAdapter` **Read**

C++ Syntax HRESULT get_ForceDeEmbedENRAdapter(VARIANT_BOOL* on);
HRESULT put_ForceDeEmbedENRAdapter(VARIANT_BOOL on);

Interface INoiseCal2

Last Modified:

26-Oct-2009 MX New topic

ForceDeEmbedSensorAdapter Property

Description Set and read the state of power sensor adapter de-embedding. [Learn more](#).

VB Syntax `noiseCal.ForceDeEmbedSensorAdapter = value`

Variable [\(Type\)](#) - Description

noiseCal A [NoiseCal](#) (object)

value (**boolean**) - Power sensor adapter de-embed state.

False - Do not Force de-embedding.

True - Force de-embedding.

Return Type Boolean

Default False

Examples

```
noiseCal.ForceDeEmbedSensorAdapter = False 'Write
```

```
AdapterDembed = noiseCal.ForceDeEmbedSensorAdapter 'Read
```

C++ Syntax HRESULT get_ForceDeEmbedSensorAdapter(VARIANT_BOOL* on);
HRESULT put_ForceDeEmbedSensorAdapter(VARIANT_BOOL on);

Interface INoiseCal2

Last Modified:

26-Oct-2009 MX New topic

Format Property (marker)

Description Sets (or returns) the format of the marker.

VB Syntax *mark.Format = value*

Variable [\(Type\)](#) - Description

mark A [Marker](#) (object)

value (enum `NAMarkerFormat`) - Choose from:

0 - `naMarkerFormat_LinMag`

1 - `naMarkerFormat_LogMag`

2 - `naMarkerFormat_Phase`

3 - `naMarkerFormat_Delay`

4 - `naMarkerFormat_Real`

5 - `naMarkerFormat_Imaginary`

6 - `naMarkerFormat_SWR`

7 - `naMarkerFormat_LinMagPhase`

8 - `naMarkerFormat_LogMagPhase`

9 - `naMarkerFormat_RealImaginary`

10 - `naMarkerFormat_ComplexImpedance`

11 - `naMarkerFormat_ComplexAdmittance`

12 - `naMarkerFormat_Kelvin`

13 - `naMarkerFormat_Fahrenheit`

14 - `naMarkerFormat_Celsius`

Return Type `NAMarkerFormat`

Default 1 - `naMarkerFormat_LogMag`

Examples `mark.Format = naMarkerFormat_SWR 'Write`

`fmt = mark.Format 'Read`

C++ Syntax HRESULT get_Format(tagNAMarkerFormat *pVal)
HRESULT put_Format(tagNAMarkerFormat newVal)

Interface IMarker

Last Modified:

1-Oct-2007 Added temperature formats

FormatUnit Property

Description Sets and returns the units for the specified data format. Measurements with display formats other than those specified are not affected.

VB Syntax *meas.FormatUnit (format) = value*

Variable [\(Type\)](#) - Description

meas A [Measurement](#) (object)

format (enum **NADataFormat**) - Choose:

1 - **naDataFormat_LogMag**

value (enum **naFormatUnit**) - Choose from:

0 - naFormatUnit_dBm Units are displayed in dBm. 0 dBm = 0.001 watt

1 - naFormatUnit_dBmV Units are displayed in dBmV. 0 dBmV = 0.001 volt

Return Type Enum

Default **0 - naFormatUnit_dBm**

Examples `meas.FormatUnit(1) = naFormatUnit_dBmV 'Write`

`units = meas.FormatUnit(1) 'Read`

C++ Syntax HRESULT put_FormatUnit(tagDataFormat format, tagFormatUnit unit)

HRESULT get_FormatUnit(tagDataFormat format, tagFormatUnit* unit)

Interface IMeasurement9

Last Modified:

26-Aug-2008 MX New topic

Frequency Property

Description Sets group delay aperture using a fixed frequency range.

VB Syntax *gdAperture.Frequency = value*

Variable [\(Type\)](#) - [Description](#)

gdAperture A [GroupDelayAperture](#) (**object**)

value (**Double**) Frequency range (in Hz) to use for the aperture setting.

Return Type Double

Default Frequency range that equates to 11 points.

This can be changed to two points with a [preference setting](#).

Examples `gdAperture.Frequency = 1e6 'Write`

`aperture = gdAperture.Frequency 'Read`

C++ Syntax HRESULT get_Frequency(double Frequency *pVal)

HRESULT put_Frequency(double Frequency newVal)

Interface IGroupDelayAperture

Last Modified:

23-Feb-2010 MX New topic

Frequency Property

Description	Sets or returns the frequency associated with a Power Sensor CalFactor Segment or Sets or returns the frequency associated with a Power Loss Segment.
VB Syntax	<i>object</i> . Frequency = <i>value</i>
Variable	(Type) - Description
<i>object</i>	One of the following objects: PowerSensorCalFactorSegment PowerSensorCalFactorSegmentPMAR PowerLossSegment PowerLossSegmentPMAR
<i>value</i>	(double) – Frequency in units of Hz. This can be any non-negative value (limited by the maximum value of double).
Return Type	Double
Default	0
Examples	<pre>seg.Frequency = 6e9 'Write freq = seg.Frequency 'Read</pre>
C++ Syntax	HRESULT put_Frequency(double newVal); HRESULT get_Frequency(double *pVal);
Interface	One of the above objects.

Last Modified:

25-Aug-2009 Added PMAR

FrequencyCenter Property

Description Sets and returns the center frequency of the main tones. Use with IMD sweep types:

- `naIMDToneCWSweep`
- `naIMDTonePowerSweep`
- `naIMDDeltaFrequencySweep`

VB Syntax `object.FrequencyCenter = value`

Variable [\(Type\)](#) - Description

object A [SweptIMD](#) or [IMSpectrum](#) Object

value (Double) Tone center frequency in Hz. Both the F1 and F2 tones MUST be within the frequency range of the PNA.

Return Type Double

Default 1.0 GHz

Examples

```
imd.FrequencyCenter = 2e9 'Write
```

```
value = imd.FrequencyCenter 'Read
```

C++ Syntax HRESULT get_FrequencyCenter(double *pVal)
HRESULT put_FrequencyCenter(double *pVal)

Interface ISweptIMD
IIMSpectrum

Last Modified:

19-Aug-2008 MX New topic

FrequencyCenterCenter Property

Description Sets and returns the sweep center frequency when sweeping the main tones. Use with sweep type = **naIMDToneCenterFreqSweep**.

VB Syntax *imd.FrequencyCenterCenter = value*

Variable [\(Type\)](#) - Description

imd A [SweptIMD](#) Object

value (Double) Center frequency in Hz. The F1 and F2 tones MUST be within the frequency range of the PNA.

Return Type Double

Default 13.255 GHz

Examples

```
imd.FrequencyCenterCenter = 10e9 'Write  
value = imd.FrequencyCenterCenter 'Read
```

C++ Syntax HRESULT get_FrequencyCenterCenter(double *pVal)
HRESULT put_FrequencyCenterCenter(double newVal)

Interface ISweptIMD

Last Modified:

19-Aug-2008 MX New topic

FrequencyCenterSpan Property

Description Sets and returns the frequency span when sweeping the main tones. Use with sweep type = **naIMDToneCenterFreqSweep**.

VB Syntax *imd.FrequencyCenterSpan = value*

Variable [\(Type\)](#) - Description

imd A [SweptIMD](#) Object

value (Double) Frequency span in Hz. The F1 and F2 tones MUST be within the frequency range of the PNA.

Return Type Double

Default 26.489 GHz

Examples

```
imd.FrequencyCenterSpan = 10e9 'Write  
value = imd.FrequencyCenterSpan 'Read
```

C++ Syntax HRESULT get_FrequencyCenterSpan(double *pVal)
HRESULT put_FrequencyCenterSpan(double newVal)

Interface ISweptIMD

Last Modified:

19-Aug-2008 MX New topic

FrequencyCenterStart Property

Description Sets and returns the start frequency when sweeping the main tones. Use with sweep type = **naIMDToneCenterFreqSweep**.

VB Syntax *imd.FrequencyCenterStart = value*

Variable [\(Type\)](#) - Description

imd A [SweptIMD](#) Object

value (Double) Start frequency in Hz. The F1 and F2 tones MUST be within the frequency range of the PNA.

Return Type Double

Default 10.5 MHz

Examples

```
imd.FrequencyCenterStart = 20e6 'Write  
value = imd.FrequencyCenterStart 'Read
```

C++ Syntax HRESULT get_FrequencyCenterStart(double *pVal)
HRESULT put_FrequencyCenterStart(double newVal)

Interface ISweptIMD

Last Modified:

19-Aug-2008 MX New topic

FrequencyCenterStop Property

Description Sets and returns the stop frequency when sweeping the main tones. Use with sweep type = **naIMDToneCenterFreqSweep**.

VB Syntax *imd.FrequencyCenterStop = value*

Variable [\(Type\)](#) - Description

imd A [SweptIMD](#) Object

value (Double) Stop frequency in Hz. The F1 and F2 tones MUST be within the frequency range of the PNA.

Return Type Double

Default 26.4995 MHz

Examples

```
imd.FrequencyCenterStop = 20e9 'Write  
value = imd.FrequencyCenterStop 'Read
```

C++ Syntax HRESULT get_FrequencyCenterStop(double *pVal)
HRESULT put_FrequencyCenterStop(double newVal)

Interface ISweptIMD

Last Modified:

19-Aug-2008 MX New topic

FrequencyOffsetDivisor Property **Superseded**

Description	This method is replaced by properties on the FOMRange Object . Specifies (along with FrequencyOffsetMultiplier) the value to multiply by the stimulus. See other Frequency Offset properties
VB Syntax	<i>object</i> . FrequencyOffsetDivisor = <i>value</i>
Variable	(Type) - Description
<i>object</i>	Channel (object) or CalSet (object) - Read-only property
<i>value</i>	(Double) - Divisor value. Range is 1 to 1000
Return Type	Double
Default	1
Examples	<pre>chan.FrequencyOffsetDivisor = 2 'Write</pre> <pre>fOffsetDiv = chan.FrequencyOffsetDivisor 'Read</pre>
C++ Syntax	HRESULT get_FrequencyOffsetDivisor(double*pval) HRESULT put_FrequencyOffsetDivisor(double newVal)
Interface	IChannel2 CalSet3

FrequencyOffsetFrequency Property **Superseded**

Description	This method is replaced by properties on the FOMRange Object . Specifies an absolute offset frequency in Hz. For mixer measurements, this would be the LO frequency. See other Frequency Offset properties.
VB Syntax	<i>object</i> . FrequencyOffsetFrequency = <i>value</i>
Variable	(Type) - Description
<i>object</i>	Channel (object)
	or
	CalSet (object) - Read-only property
<i>value</i>	(Double) - Offset value. Range is +/- 1000 GHz. (Offsets can be positive or negative.)
Return Type	Double
Default	0 Hz
Examples	<pre>chan.FrequencyOffsetFrequency = 2 'Write</pre> <pre>fOffsetFreq = chan.FrequencyOffsetFrequency 'Read</pre>
C++ Syntax	HRESULT get_FrequencyOffsetFrequency(double*pval) HRESULT put_FrequencyOffsetFrequency(double newVal)
Interface	IChannel2 CalSet3

FrequencyOffsetMultiplier Property **Superseded**

Description	This method is replaced by properties on the FOMRange Object . Specifies (along with FrequencyOffsetDivisor) the value to multiply by the stimulus. See other Frequency Offset properties.
VB Syntax	<i>object</i> . FrequencyOffsetMultiplier = <i>value</i>
Variable	(Type) - Description
<i>object</i>	Channel (object) or CalSet (object) - Read-only property
<i>value</i>	(Double) - Multiplier value. Range is 1 to 1000
Return Type	Double
Default	1
Examples	<pre>chan.FrequencyOffsetMultiplier = 2 'Write</pre> <pre>fOffsetMult = chan.FrequencyOffsetMultiplier 'Read</pre>
C++ Syntax	HRESULT get_FrequencyOffsetMultiplier (double*pval); HRESULT put_FrequencyOffsetMultiplier (double newVal);
Interface	IChannel2 CalSet3

Write/Read

FrequencyOffsetRangeForCalComputations Property

Description Specifies the FOM frequency range to use when performing calibration.

VB Syntax `pref.FrequencyOffsetRangeForCalComputations = value`

Variable [\(Type\)](#) - Description

pref A [Preferences](#) (object)

value (Enum as [NACalFOMRange](#)) - Choose from:

0 - naCalFOMRangeAuto - All other calibration situations.

1 - naCalFOMRangePrimary - Used for calibrating at the mmWave frequencies when NOT using a test set. [Learn more.](#)

Return Type Enum

Default [naCalFOMRangeAuto](#)

Examples

```
pref.FrequencyOffsetRangeForCalComputations = naCalFOMRangePrimary 'Write  
calPref = pref.FrequencyOffsetRangeForCalComputations 'Read
```

C++ Syntax

```
HRESULT get_FrequencyOffsetRangeForCalComputations(tagNACalFOMRange *  
val);  
  
HRESULT put_FrequencyOffsetRangeForCalComputations((tagNACalFOMRange val);
```

Interface [IPreferences10](#)

Last Modified:

16-Sep-2009 MX New topic

FrequencyOffsetCWOverride Property **Superseded**

Description	This method is replaced by properties on the FOMRange Object .
	Establishes a fixed (CW) stimulus frequency while measuring the Response over a swept frequency range. For example, a fixed-frequency PNA stimulus may be applied to the RF input of a mixer whose local oscillator (LO) is being swept. Because the IF output of the mixer will be swept, the PNA receivers must also be swept.
	See other Frequency Offset properties.
VB Syntax	<i>object</i> . FrequencyOffsetCWOverride = <i>value</i>
Variable	(Type) - Description
<i>object</i>	Channel (object)
	or
	CalSet (object) - Read-only property
<i>value</i>	(Enum as NaStates) - Choose from:
	naOFF (0) - Turns CW override OFF
	naON (1) - Turns CW override ON
Return Type	Enum
Default	0 Hz
Examples	<pre>chan.FrequencyOffsetCWOverride = 1 'Write</pre>
	<pre>fOffsetOV = chan.FrequencyOffsetCWOverride 'Read</pre>
C++ Syntax	HRESULT get_FrequencyOffsetCWOverride (tagNAStates *pstate) HRESULT put_FrequencyOffsetCWOverride (tag NAStates newState)
Interface	IChannel2 CalSet3

FrequencyOffsetState Property **Superseded**

Description	This method is replaced by properties on the FOMRange Object . Enables Frequency Offset on ALL measurements that are present on the active channel. This immediately causes the source and receiver to tune to separate frequencies. The receiver frequencies are specified with other channel and offset settings. To make the stimulus settings, use Channel Start, Stop Frequency properties. See other Frequency Offset properties. Tip: To avoid unnecessary errors, first make other frequency offset settings. Then turn Frequency Offset ON.
VB Syntax	<i>object</i> . FrequencyOffsetState = <i>value</i>
Variable	(Type) - Description
<i>object</i>	Channel (object) or CalSet (object) - Read-only property
<i>value</i>	(Enum as NaStates) - Choose from: naOFF (0) - Turns Frequency Offset OFF naON (1) - Turns Frequency Offset ON
Return Type	Enum
Default	naOFF (0)
Examples	<pre>chan.FrequencyOffsetState = naON 'Write</pre> <pre>Foffset = chan.FrequencyOffsetState 'Read</pre>
C++ Syntax	HRESULT FrequencyOffsetState (tag NAStates *pState); HRESULT FrequencyOffsetState (tag NAStates newState)
Interface	IChannel2 CalSet3

FrequencyType Property

Description	Sets and returns the frequency range to use for receiver leveling. On the user interface, this is the "Receiver frequency is determined by:" setting.
VB Syntax	<i>RxLevel.FrequencyType(srcPort) = value</i>
Variable	(Type) - Description
<i>RxLevel</i>	A ReceiverLeveling Object
<i>srcPort</i>	(Long Integer) Source port for Receiver Leveling.
	<p>Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use chan.getPortNumber to translate the string into a port number. To learn more see Remotely Specifying a Source Port</p>
<i>value</i>	(Enum) Frequency range. Choose from: <ul style="list-style-type: none"> naAutoFrequency (0) - always uses the frequency range that is assigned to the measurement receiver. naInputFrequency (3) - Mixer/Converter input frequency range. naOutputFrequency (4) - Mixer/Converter input frequency range. naReceiverFrequency (1) - FOM Receiver frequency range. naSourceFrequency (2) - FOM Source frequency range
Return Type	Enum
Default	naAutoFrequency (0)
Examples	<pre>rxLevel.FrequencyType (1) = naAutoFrequency ' Write value = rxLevel.FrequencyType 2 ' Read</pre>
C++ Syntax	<pre>HRESULT get_FrequencyType(long port, tagNAFrequencyType* pVal); HRESULT put_FrequencyType(long port, tagNAFrequencyType newVal);</pre>
Interface	IRxLevelingConfiguration3

Last Modified:

8-Mar-2012 MX New topic

Gain Property

Description Returns the Gain result of the PNOP marker search.

$\text{Pnop Gain} = \text{Pnop Out} - \text{Pnop In}$.

VB Syntax `gain = pnop.Gain`

Variable [\(Type\)](#) - Description

`gain` (**double**) - Variable to store returned value

`pnop` A [PNOP](#) (**object**)

Return Type Double

Default Not applicable

Examples `gain = pnop.Gain 'Read`

[See example program](#)

C++ Syntax HRESULT get_Gain(double* pNewVal)

Interface IPNOP

Last Modified:

19-Feb-2010 MX New topic

GainLinear Property

Description Returns the Linear Gain result of a PSat marker search.

Gain Linear = Marker 1 - Y-axis value MINUS X-axis value

VB Syntax *gainLin* = *pSat*.GainLinear

Variable [\(Type\)](#) - Description

gainLin (double) - Variable to store returned value

pSat A [PSaturation](#) (object)

Return Type Double

Default Not applicable

Examples `gainLin = pSat.GainLinear 'Read`

[See example program](#)

C++ Syntax HRESULT get_GainLinear(double* pNewVal)

Interface IPSaturation

Last Modified:

19-Feb-2010 MX New topic

GainMax Property

Description Returns the GainMax result of the PNOP or PSAT marker search.

$$\text{Gain Max} = \text{PMax Out} - \text{PMax In}$$

VB Syntax *gainMax* = *pMarker*.**GainMax**

Variable (Type) - Description

gainMax (double) - Variable to store returned value

pMarker A [PNOP](#) (object) or [PSaturation](#) (Object)

Return Type Double

Default Not applicable

Examples `gainMax = pMarker.GainMax 'Read`

[See example program](#)

C++ Syntax HRESULT get_GainMax(double* pNewVal)

Interface IPNOP or IPSaturation

Last Modified:

19-Feb-2010 MX New topic

GainSaturation Property

Description Returns the GainSaturation result of the PSAT marker search.

Gain Sat = Psat Out - Psat In

VB Syntax *GainSat* = *pSat*.**GainSaturation**

Variable [\(Type\)](#) - Description

GainSat (double) - Variable to store returned value

pSat A [PSaturation](#) (Object)

Return Type Double

Default Not applicable

Examples `GainSat = pSat.GainSaturation 'Read`

[See example program](#)

C++ Syntax HRESULT get_GainSaturation(double* pNewVal)

Interface IPSaturation

Last Modified:

22-Feb-2010 MX New topic

GeneratedCalsets Property

Description	Returns the cal set names that were produced by the cal all session.
VB Syntax	<i>csets</i> = <i>calAll</i> . GeneratedCalsets
Variable	(Type) - Description
<i>csets</i>	(Variant Array) Variable to store the returned cal set names.
<i>calAll</i>	A CalibrateAllChannels (object)
Return Type	Variant Array
Default	Not Applicable
Examples	<pre>csets = calAll.GeneratedCalsets</pre> <pre>See example program</pre>
C++ Syntax	HRESULT get_GeneratedCalsets (VARIANT*, propNames);
Interface	CalibrateAllChannels

Last modified:

28-Mar-2012 New topic

FrequencySpan Property

Description Sets or returns the frequency span of the channel.

Sets or returns the frequency span of the segment.

VB Syntax *object*.FrequencySpan = *value*

Variable [\(Type\)](#) - Description

object A Channel (**object**)

or

A Segment (**object**)

value (**double**) - Frequency span in Hertz. Choose any number between the **minimum** and **maximum** frequencies of the analyzer.

Return Type Double

Default Full frequency span of the analyzer

Examples

```
chan.FrequencySpan = 4.5e9 'sets the frequency span of a linear sweep  
for the channel object -Write
```

```
freqspan = chan.FrequencySpan 'Read
```

C++ Syntax HRESULT get_FrequencySpan(double *pVal)
HRESULT put_FrequencySpan(double newVal)

Interface IChannel
ISegment

Shape Property

Description Specifies the shape of the gate filter.

VB Syntax `gat.Shape = value`

Variable [\(Type\)](#) - Description

gat A Gating (**object**)

value (**enum** NAGateShape) - Choose from:

0 - naGateShapeMaximum

1 - naGateShapeWide

2 - naGateShapeNormal

3 - naGateShapeMinimum

Return Type NAGateShape

Default 2 - Normal

Examples `gat.Shape = naGateShapeMaximum 'Write`

`filterShape = gat.Shape 'Read`

C++ Syntax HRESULT get_Shape(tagNAGateShape *pVal)
HRESULT put_Shape(tagNAGateShape newVal)

Interface IGating

Type (gate) Property

Description	Specifies the type of gate filter used.
VB Syntax	<i>gat.Type</i> = <i>value</i>
Variable	(Type) - Description
<i>gat</i>	A Gating (object)
<i>value</i>	(enum NAGateType) - Choose from: 0 - naGateTypeBandpass - Includes (passes) the range between the start and stop times. 1 - naGateTypeNotch - Excludes (attenuates) the range between the start and stop times.
Return Type	NAGateType
Default	Bandpass
Examples	<pre>gate.Type = naGateTypeNotch 'Write</pre> <pre>filterType = gate.Type 'Read</pre>
C++ Syntax	HRESULT get_Type(tagNAGateType *pVal) HRESULT put_Type(tagNAGateType newVal)
Interface	IGating

GPIBAddress Property

Description Sets and returns the PNA GPIB address on the talker/listener bus.

VB Syntax *app.GPIBAddress (bus) = value*

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

bus (Short Integer) GPIB bus. MUST be set to 0 - the talker/listener bus.

value (Short Integer) GPIB Address on the PNA. Choose a value between 0 and 30.

Return Type Short Integer

Default 16

Examples `address=app.GPIBAddress(0) 'Read`

`app.GPIBAddress(0)=16 'Write`

C++ Syntax HRESULT get_GPIBAddress(short busIndex, short* address);

HRESULT put_GPIBAddress(short busIndex,short address);

Interface IApplication8

Last Modified:

16-Mar-2011 Add talker/listener

GPIBMode Property

Description Changes the analyzer to a GPIB system controller or a talker/listener on the bus. The analyzer must be the controller if you want to use it to send commands to other instruments. The analyzer must be a talker/listener if you want to send it commands from another PC.

Note: This command has no affect in PNAs with dedicated Controller and Talker/Listener GPIB connectors.

VB Syntax *app.GPIBMode value*

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

value (enum **NAGPIBMode**) -Choose either:
0 - naTalkerListener - the analyzer is a talker / listener
1 - naSystemController - the analyzer is the system controller

Return Type Long Integer

Default 0 - naTalkerListener

Examples

```
app.GPIBMode = naTalkerListener 'Write  
mode = app.GPIBMode 'Read
```

C++ Syntax HRESULT get_GPIBMode(tagGPIBModeEnum* eGpibMode)
HRESULT put_GPIBMode(tagGPIBModeEnum eGpibMode)

Interface IApplication

Last Modified:

10-Nov-2008 Removed link to obsolete topic.

GPIBPortCount Property

Description	Returns the number of GPIB ports that are present on the PNA rear-panel.
VB Syntax	<i>value</i> = <i>cap</i> . GPIBPortCount
Variable	(Type) - Description
<i>value</i>	(Long) - Variable to store the returned integer value of the number of GPIB ports.
<i>cap</i>	A Capabilities (object)
Return Type	Long
Default	Not Applicable
Examples	<pre>value = cap.GPIBPortCount 'Read</pre>
C++ Syntax	HRESULT get_GPIBPortCount(long * gpibPorts);
Interface	ICapabilities3

Last Modified:

23-Oct-2008 Removed 1.1 GHz description

Grid Property

Description Set and return the inner lines of all grid in all windows, and the grid frame in inactive windows for the PNA display or hardcopy print.

VB Syntax *colors*.**Grid** = *value*

Variable [\(Type\)](#) - Description

colors A [ComColors](#) (object)

value **(Long Integer)** - RGB color of the Grid pen.

Convert the three RGB colors to an integer as follows:

```
RGB = R+(G*2^8)+(B*2^16)
```

To find the three RGB values from the [Display Colors dialog](#), click **Change Color**, then **Define Custom Color**.

Return Type Long

Default Display = 175,175,175

Print = 0,0,0 (Black)

Examples

```
R = 10
G = 10
B = 10
RGB = R+(G*2^8)+(B*2^16)
colors.Grid = RGB 'Write
color = colors.Grid 'Read
```

C++ Syntax HRESULT get_Grid(long* pVal);

HRESULT put_Grid(long newVal);

Interface IComColors

Last Modified:

7-Aug-2009 MX New topic

GridLineType Property

Description	Set and return whether the grid lines are displayed in solid or dotted lines for all open windows. Grid lines are returned to solid when the PNA is Preset.
VB Syntax	<i>app</i> .GridLineType = <i>value</i>
Variable	(Type) - Description
<i>app</i>	A Application Object (object)
<i>value</i>	(Enum as naLineType) - Choose from: 0 - naLineTypeSolid 1 - naLineTypeDotted
Return Type	Enum
Default	naLineTypeSolid
Examples	<pre>app.GridLineType = naLineTypeSolid 'Write grid = app.GridLineType 'Read</pre>
C++ Syntax	HRESULT get_GridLineType(tag naLineType* pVal); HRESULT put_GridLineType(tag naLineType newVal);
Interface	IApplication

Last Modified:

16-Mar-2010 MX New topic

HandshakeEnable Property

Description Turns handshake ON / OFF.

To enable handshake, the main trigger enable must ALSO be set using [Enable](#).

When ON, PNA acquisition waits indefinitely for the input line to be asserted before continuing with the acquisition.

Note: Use on PNA-X ONLY. Other models do NOT have an Aux Input.

VB Syntax `auxTrig.HandshakeEnable = state`

Variable [\(Type\)](#) - Description

`auxTrig` An [AuxTrigger](#) (object)

`state` (boolean) -

True - Handshake enabled

False - Handshake NOT enabled

Return Type Boolean

Default False

Examples `auxTrig.HandshakeEnable = True 'Write`

`value = auxTrig.HandshakeEnable 'Read`

C++ Syntax `HRESULT get_HandshakeEnable(VARIANT_BOOL * enable);`
`HRESULT put_HandshakeEnable(VARIANT_BOOL enable);`

Interface IAuxTrigger

Last Modified:

5-Sep-2008 Added Note
 14-Dec-2006 MX New topic

HasItem Property

Description	Returns a value indicating whether the specified external devices is configured.
VB Syntax	<i>value</i> = <i>extDevices</i> . HasItem (<i>name</i>)
Variable	(Type) - Description
<i>value</i>	(Boolean) Variable to store one of the following returned values. False - Item (name) is NOT present in the External Devices collection. True - Item (name) IS present in the External Devices collection.
<i>extDevices</i>	An ExternalDevices (collection)
<i>name</i>	(String) Name of External Device for which to search the collection.
Return Type	Boolean
Default	Not Applicable
Examples	<pre>has = extDevices.HasItem('mysource') 'Read</pre>
C++ Syntax	HRESULT get_HasItem(VARIANT Index, BOOL *pVal);
Interface	IExternalDevices

Last Modified:

27-Nov-2012	Fixed
31-Jul-2009	MX New topic

HighAmplitude Property

Description Sets and returns the High amplitude (voltage) of the pulse generator.

VB Syntax *extPulseGen.HighAmplitude* = *value*

Variable [\(Type\)](#) - Description

extPulseGen An [ExternalPulseGenerator](#) (object)

value (**Double**) Pulse Generator high amplitude voltage.

Return Type Double

Default 5

Examples `extPulseGen.HighAmplitude = 4 'Write`

`hi = extPulseGen.HighAmplitude 'Read`

C++ Syntax HRESULT get_HighAmplitude (double *pValue)

HRESULT put_HighAmplitude (double newVal)

Interface IExternalPulseGenerator

Last Modified:

15-Feb-2012 New topic

HighestOrderProduct Property

Description	Returns the highest order product that can be measured by SweptIMD.
VB Syntax	<i>value</i> = <i>imd</i> . HighestOrderProduct
Variable	(Type) - Description
<i>value</i>	(Long Integer) Variable in which to store the returned value.
<i>imd</i>	A SweptIMD Object
Return Type	Integer
Default	Always returns 9
Examples	<pre>value = imd.HighestOrderProduct 'Read</pre>
C++ Syntax	HRESULT get_HighestOrderProduct(long *pVal)
Interface	ISweptIMD

Last Modified:

19-Sep-2008 MX New topic

ID Property

Description	Returns the test set ID number. For GPIB testsets, the ID is equivalent to the GPIB address. For testset I/O testsets, the ID is the base address of the testset (0 for the first testset, 1 for the second, and so on).
VB Syntax	<i>value</i> = <i>tset</i> . ID
Variable	(Type) - Description
<i>value</i>	(Long) variable to store the returned information.
<i>testsets(1)</i>	A TestsetControl object.
	OR
	An E5091Testset object.
Return Type	Long Integer
Default	Not Applicable
Examples	<pre>value = testset1.ID</pre> See E5091A Example Program See External Testset Program
C++ Syntax	HRESULT get_ID(long *idNumber);
Interface	ITestsetControl IE5091Testset

Read-only

IDString Property

Description Returns the ID of the analyzer, including the Model number, Serial Number, and the Software revision number.

Note: Beginning with Rev 6.01, this command now returns the software revision with 6 digits instead of 4. For example, A.06.01.02.

VB Syntax *value* = *app*.IDString

Variable [\(Type\)](#) - Description

app An [Application](#) (**object**)

value (**string**) - variable to contain the returned ID string

Return Type String

Default Not Applicable

Examples `id = app.IDString`

C++ Syntax HRESULT IDString(BSTR* IDString)

Interface IApplication

IFBandwidthOption Property

Description	Enables the IFBandwidth to be set on individual sweep segments. This property must be set True before <code>seg.IFBandwidth = value</code> is sent. Otherwise, this command will be ignored.
VB Syntax	<code>segs.IFBandwidthOption = value</code>
Variable	(Type) - Description
<i>segs</i>	A Segments collection (object)
<i>value</i>	(boolean) True - Enables variable IFBandwidth setting for segment sweep False - Disables variable IFBandwidth setting for segment sweep
Return Type	Boolean
Default	False
Examples	<pre>segs.IFBandwidthOption = True 'Write IFOption = IFBandwidthOption 'Read</pre>
C++ Syntax	HRESULT get_IFBandwidthOption(VARIANT_BOOL *pVal) HRESULT put_IFBandwidthOption(VARIANT_BOOL newVal)
Interface	ISegments

IFBandwidth Property

Description Sets or returns the IF Bandwidth of the channel.

Sets or returns the IF Bandwidth of the segment.

Returns the IF Bandwidth used in the Cal Set

VB Syntax *object*.**IFBandwidth** = *value*

Variable [\(Type\)](#) - Description

object Channel (**object**)

or

Segment (**object**)

or

CalSet (**object**) - Read-only property

value (**double**) - IF Bandwidth in Hz. The list of valid IF Bandwidths is different depending on the PNA model. ([Click to see the lists.](#)) If an invalid number is specified, the analyzer will round up to the closest valid number.

Return Type Double

Default Varies with PNA model.

Examples

```
chan.IFBandwidth = 3e3 'sets the IF Bandwidth of for the channel object
to 3 kHz. -Write
seg.IFBandwidth = 5 'sets the IF Bandwidth of the segment to 5 Hz. -
Write
```

```
ifbw = chan.IFBandwidth -Read
```

C++ Syntax HRESULT get_IFBandwidth(double *pVal);

HRESULT put_IFBandwidth(double newVal);

Interface IChannel
ISegment

|CalSet3

IFBW Property

Description	Sets and returns the IFBW for a Cal All calibration. Learn more about this setting.
VB Syntax	<i>calAll</i> . IFBW = <i>value</i>
Variable	(Type) - Description
<i>calAll</i>	A CalibrateAllIFBW (object)
<i>value</i>	(Double) IFBW in Hz.
Return Type	Variant
Default	1 kHz
Examples	<pre>calAll.IFBW = 10e3 'sets IFBW to 10 kHz</pre> <pre>ifbw = calAll.IFBW 'returns the ifbw setting</pre>
C++ Syntax	HRESULT get_IFBW (Double *Val) HRESULT put_IFBW (Double newVal)
Interface	ICalibrateAllChannels

Last modified:

4-Jan-2012 New topic

IFBWList Property

Description Returns a list of supported IFBW values.

VB Syntax *value* = *cap*.**IFBWList**

Variable [\(Type\)](#) - Description

value (Variant) - Variable to store the returned array of IFBW values

cap A [Capabilities](#) (Object)

Return Type Variant array

Default Not Applicable

Examples

```
'Read the supported IFBW values

Set app =
CreateObject("AgilentPNA835x.Application")

Set cap = app.Capabilities

list=cap.IFBWList

For i = 0 To UBound(list)

    msg = msg & list(i) & vbCrLf

Next

MsgBox msg
```

C++ Syntax HRESULT get_IFBWList(Variant *value);

Interface ICapabilities8

Last Modified:

23-May-2011 MX New topic

IFDenominator Property

Description Sets or returns the denominator value of the IF Fractional Multiplier.

Only applies to 2 stage mixers.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

VB Syntax *conv*.IFDenominator = *value*

Variable [\(Type\)](#) - Description

conv A [Converter](#) (object)

value (**long**) IF Denominator value.

Return Type Long

Default 1

Examples `Print mixer.IFDenominator 'prints the value of the IFDenominator`

C++ Syntax HRESULT get_IFDenominator(long *pVal)

HRESULT put_IFDenominator(long newVal)

Interface IConverter4

Last Modified:

29-Oct-2010 MX New topic

IFNumerator Property

Description Sets or returns the numerator value of the IF Fractional Multiplier.

Only applies to 2 stage mixers.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

VB Syntax *conv*.IFNumerator = *value*

Variable [\(Type\)](#) - Description

conv A [Converter](#) (object)

value (**Long**) IF Numerator value.

Return Type long

Default 1

Examples `Print mixer.IFNumerator 'prints the value of the IFNumerator`

C++ Syntax HRESULT get_IFNumerator(long *pVal)

HRESULT put_IFNumerator(long newVal)

Interface IConverter4

Last Modified:

29-Oct-2010 New topic

IFFrequency Property

Description	Sets and returns the IF frequency for ALL receiver paths being used for the specified channel. To set this frequency, IFFrequencyModeProperty must be set to OFF (Manual).
VB Syntax	<i>IfConfig</i> . IFFrequency = <i>value</i>
Variable	(Type) - Description
<i>IfConfig</i>	An IFConfiguration (object)
<i>value</i>	(double) IF Frequency. Use MaximumIFFrequency and MinimumIFFrequency to determine the range of value for this command.
Return Type	Double
Default	9 MHz
Examples	<pre>IfConfig.IFFrequency = 9.3e6</pre>
C++ Syntax	HRESULT get_IFFrequency (double *pVal); HRESULT put_IFFrequency (double pVal);
Interface	IFConfiguration3

Last Modified:

25-Aug-2010 Updated for DSP 5
18-Jun-2007 MX New topic

IFFrequencyMode Property

Description	Sets and returns method for specifying the way the IF Frequency is determined.
VB Syntax	<i>IfConfig</i> . IFFrequencyMode = <i>value</i>
Variable	(Type) - Description
<i>IfConfig</i>	An IFConfiguration (object)
<i>value</i>	(enum as NAModes) IF Frequency mode. Choose from: 0 - naAUTO The PNA determines the setting for the IF frequency. The IF frequency is based on many PNA settings, including measurement frequency. Therefore, it is NOT possible to read the IF frequency that is being used. 1 - naMANUAL (use IFFrequencyProperty to manually set frequency.
Return Type	Enum
Default	0 naAuto
Examples	<pre>IfConfig.IFFrequencyMode = naMANUAL</pre>
C++ Syntax	<pre>HRESULT get_IFFrequencyMode (tagNAModes* pdspMode); HRESULT put_IFFrequencyMode (tagNAModes* pdspMode);</pre>
Interface	IFConfiguration3

Last Modified:

19-Oct-2010 Added Auto note
18-Jun-2007 MX New topic

IFSideband Property

Description When two LOs are used, sets or returns whether to select the sum or difference for the IF1 product. (Input + or - LO1 = IF1)



- This setting corresponds to the   buttons on LO1.
- Also set [OutputSideband](#) to LOW or HIGH to determine the output frequency of the mixer.
- This setting is ignored when ONE LO is used.
- If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

Note: There is also an [IFSideband_Property](#) on the Mixer Interface.

VB Syntax `obj.IFSideband =value`

Variable [\(Type\)](#) - Description

obj A [Converter Object](#)

value (enum as **ConverterSideBand**) - Choose from:

0 or **naLowSide** Minus (-) on the Mixer setup dialog

1 or **naHighSide** Plus (+) on the Mixer setup dialog

Return Type Enum as ConverterSideBand

Default 0 - **naLowSide**

Examples `conv.IFSideband = naLowSide`

C++ Syntax `HRESULT get_IFSideband(ConverterSideBand *pVal)`

`HRESULT put_IFSideband(ConverterSideBand newVal)`

Interface IConverter

Last Modified:

26-Mar-2009 New topic

IFSideband Property

Description When two LOs are used, sets or returns whether to select the sum or difference for the IF1 product. (Input + or - LO1 = IF1)



- This setting corresponds to the   buttons on LO1.
- Also set [OutputSideband](#) to LOW or HIGH to determine the output frequency of the mixer.
- This setting is ignored when ONE LO is used.
- If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

Note: There is also an [IFSideband_Property](#) on the Converter Object.

VB Syntax `mixer.IFSideband =value`

Variable [\(Type\)](#) - [Description](#)

mixer A Mixer (**object**)

value (**enum as FCASideBand**) - Choose from:

0 or **LOW** Minus (-) on the Mixer setup dialog

1 or **HIGH** Plus (+) on the Mixer setup dialog

Return Type Enum as FCASideBand

Default 0 - LOW

Examples `Print mixer.IFSideband 'prints the value of the IFSideband`

C++ Syntax `HRESULT get_IFSideband(FCASideBand *pVal)`

`HRESULT put_IFSideband(FCASideBand newVal)`

Interface IMixer

Last Modified:

2-Oct-2008 Clarification

4-Mar-2008 Added note.

IFStartFrequency Property

Description Sets or returns the start frequency value of the mixer IF frequency.

Only applies to 2 stage mixers.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

VB Syntax *mixer*.IFStartFrequency = *value*

Variable [\(Type\)](#) - Description

mixer A [Converter](#) (object)

value **(double)** - Frequency in Hertz.

Return Type Double

Default Not Applicable

Examples `Print mixer.IFStartFrequency 'prints the value of the IFStartFrequency`

C++ Syntax HRESULT get_IFStartFrequency(double *pVal)

HRESULT put_IFStartFrequency(double newVal)

Interface IMixer

Last Modified:

4-Mar-2008 Added note.

IFStopFrequency Property

Description Sets or returns the stop frequency value of the mixer IF frequency.

Only applies to 2 stage mixers.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

VB Syntax *mixer*.IFStopFrequency = *value*

Variable [\(Type\)](#) - Description

mixer A [Converter](#) (object)

value **(double)** - IF stop frequency in Hertz.

Return Type Double

Default Not Applicable

Examples

```
Print mixer.IFStopFrequency 'prints the value of the
IFStopFrequency
```

C++ Syntax HRESULT get_IFStopFrequency(double *pVal)

HRESULT put_IFStopFrequency(double newVal)

Interface IMixer

Last Modified:

4-Mar-2008 Added note.

ImpedanceStates Method

Description Sets the number of impedance states to use during calibrated measurements.

VB Syntax `noise.ImpedanceStates = value`

Variable [\(Type\)](#) - Description

noise A [NoiseFigure](#) (object)

value (**double**) - Impedance states. Choose between 4 and the maximum number allowed by the noise tuner device. If the specified number exceeds the capability of the device, the measurement will use the maximum number of states the device allows.

Return Type Double

Default 4

Examples

```
noise.ImpedanceStates = 10 'Write
AvgNoise = noise.ImpedanceStates 'Read
```

C++ Syntax HRESULT get_ImpedanceStates(double* pVal)
HRESULT put_ImpedanceStates(double newVal)

Interface INoiseFigure

Last Modified:

29-May-2007 MN New topic

ImpulseWidth Property

Description	Sets or returns the Impulse Width of Time Domain transform windows
VB Syntax	<i>trans</i> .ImpulseWidth = <i>value</i>
Variable	(Type) - Description
<i>trans</i>	A Transform (object)
<i>value</i>	(double) - Impulse Width in seconds. Range of settings depends on the frequency range of your analyzer.
Return Type	Double
Default	.98 / Default Span
Examples	<pre>trans.ImpulseWidth = 200e-12 'sets the Impulse width of a transform window -Write IW = trans.ImpulseWidth 'Read</pre>
C++ Syntax	HRESULT get_ImpulseWidth(double *pVal) HRESULT put_ImpulseWidth(double newVal)
Interface	ITransform

IMToneIFBandwidth Property

Description Sets and returns the IF Bandwidth for measurement of the intermodulation products.

VB Syntax `imd.IMToneIFBandwidth = value`

Variable [\(Type\)](#) - Description

`imd` A [SweptIMD](#) Object

`value` (Double) IF Bandwidth in Hz. Choose from: 1 | 2 | 3 | 5 | 7 | 10 | 15 | 20 | 30 | 50 | 70 | 100 | 150 | 200 | 300 | 500 | 700 | 1k | 1.5k | 2k | 3k | 5k | 7k | 10k | 15k | 20k | 30k | 50k | 70k | 100k | 150k | 200k | 280k | 360k | 600k

[Learn more about setting IFBW for IMD.](#)

If an invalid number is specified, the analyzer will round up to the closest valid number.

Return Type Double

Default 1 kHz

Examples

```
imd.IMToneIFBandwidth = 2e3 'Write
```

```
value = imd.IMToneIFBandwidth 'Read
```

C++ Syntax HRESULT get_IMToneIFBandwidth(double *pVal)
HRESULT put_IMToneIFBandwidth(double newVal)

Interface ISweptIMD

Last Modified:

19-Aug-2008 MX New topic

InactiveLabels Property

Description Set and return the Inactive (not selected) Window Labels for the PNA display or hardcopy print.

VB Syntax *colors.InactiveLabels = value*

Variable [\(Type\)](#) - Description

colors A [ComColors](#) (object)

value **(Long Integer)** - RGB color of the Inactive Labels pen.

Convert the three RGB colors to an integer as follows:

```
RGB = R+(G*2^8)+(B*2^16)
```

To find the three RGB values from the [Display Colors dialog](#), click **Change Color**, then **Define Custom Color**.

Return Type Long

Default Display = 160,160,160

Print = 0,0,0 (Black)

Examples

```
R = 10
G = 10
B = 10
RGB = R+(G*2^8)+(B*2^16)
colors.InactiveLabels = RGB 'Write
color = colors.InactiveLabels 'Read
```

C++ Syntax HRESULT get_InactiveLabels(long* pVal);

HRESULT put_InactiveLabels(long newVal);

Interface IComColors

Last Modified:

7-Aug-2009 MX New topic

Include2ndOrderProduct Property

Description Sets and returns whether to include the second order products in the calibration. These frequencies of these products can be far from the main tones.

VB Syntax *imd.Include2ndOrderProduct = value*

Variable [\(Type\)](#) - Description

imd A [SweptIMDCal](#) (object)

value (Boolean) Choose from:

False - Do NOT include 2nd order products

True - Include 2nd order products

Return Type Boolean

Default **False** - Do NOT include 2nd order products

Examples

```
imd.Include2ndOrderProduct = true 'Write
```

```
incl = imd.Include2ndOrderProduct 'Read
```

C++ Syntax HRESULT get_Include2ndOrderProduct(VARIANT_BOOL * Val)

HRESULT put_Include2ndOrderProduct(VARIANT_BOOL newVal)

Interface ISweptIMD

Last Modified:

9-Sep-2008 MX New topic

IncludeReverseSweep Property

Description Sets whether to include SC12 sweeps during measurements.

VB Syntax *obj.IncludeReverseSweep = bool*

Variable [\(Type\)](#) - Description

obj A [Converter](#) (object)

bool **(Boolean)** -

True - Include the SC12 (reverse) sweep.

False - Do NOT Include the SC12 (reverse) sweep.

Return Type Boolean

Default True

Examples `obj.IncludeReverseSweep = True`

C++ Syntax HRESULT get_IncludeReverseSweep(VARIANT_BOOL * val);

HRESULT put_IncludeReverseSweep(VARIANT_BOOL val);

Interface IMixer12

Last Modified:

12-Feb-2010 MX New topic

IndexState Property

Description Determines the control of Material Handler connector Pin 20.

VB Syntax *handler.IndexState* = *value*

Variable [\(Type\)](#) - Description

handler **(object)** - A [Handler I/O](#) object

value **(boolean)**

False - Pin 20 is controlled by Output Port B6

True - Pin 20 is controlled by the Index signal

Return Type Boolean

Default False

Examples

```
handler.IndexState = False 'Write  
bState = handler.IndexState 'Read
```

C++ Syntax HRESULT put_IndexState (BOOL *pVal);

HRESULT get_IndexState (BOOL newVal);

Interface IHWMaterialHandlerIO2

InputA Property - **Obsolete**

Description **This property has NO replacement and no longer works correctly. (Sept. 2004)**

Sets a Port Extension value for Receiver A

VB Syntax *portExt*.InputA = *value*

Variable [\(Type\)](#) - Description

portExt A Port Extension (**object**)

value (**double**) - Port Extension value in seconds. Choose any number between **-10** and **10**

Return Type Double

Default 0

Examples

```
portExt.InputA = 10e-6 'Write
```

```
inA = portExt.InputA 'Read
```

C++ Syntax HRESULT get_InputA(double *pVal)
HRESULT put_InputA(double newVal)

Interface IPortExtension

InputB Property - **Obsolete**

Description **This property has NO replacement and no longer works correctly. (Sept. 2004)**

Sets the Port Extension value for Receiver B

VB Syntax *portExt*.InputB = *value*

Variable [\(Type\)](#) - Description

portExt A Port Extension (**object**)

value (**double**) - Port Extension value in seconds. Choose any number between **-10** and **10**

Return Type Double

Default 0

Examples

```
portExt.InputB = 10e-6 'Write
```

```
inB = portExt.InputB 'Read
```

C++ Syntax HRESULT get_InputB(double *pVal)
HRESULT put_InputB(double newVal)

Interface IPortExtension

InputC Property **Obsolete**

Description **This property has NO replacement and no longer works correctly. (Sept. 2004)**

Sets the Port Extension value for Receiver C

VB Syntax *portExt*.InputC = *value*

Variable [\(Type\)](#) - Description

portExt A Port Extension (**object**)

value (**double**) - Port Extension value in seconds. Choose any number between **-10** and **10**

Return Type Double

Default 0

Examples

```
portExt.InputC = 10e-6 'Write
```

```
inC = portExt.InputC 'Read
```

C++ Syntax HRESULT get_InputC(double *pVal)
HRESULT put_InputC(double newVal)

Interface IPortExtension

InputDenominator Property

Description Sets or returns the denominator value of the Input Fractional Multiplier.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

VB Syntax *obj*.**InputDenominator** = *value*

Variable [\(Type\)](#) - Description

obj A [Mixer Interface](#) pointer to the [Measurement](#) (object)

Or

A [Converter Object](#)

value (**Long**) - Input denominator value.

Return Type Long

Default 1

Examples

```
Print mixer.InputDenominator 'prints the value of the
InputDenominator
```

C++ Syntax HRESULT get_InputDenominator(long *pVal)

HRESULT put_InputDenominator(long newVal)

Interface IMixer

IConverter

Last Modified:

2-Feb-2009 Added converter

4-Mar-2008 Added note.

InputFixedFrequency Property

Description Sets or returns the mixer fixed Input frequency value.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

VB Syntax *obj*.InputFixedFrequency = *value*

Variable [\(Type\)](#) - Description

obj A [Mixer Interface](#) pointer to the [Measurement](#) (object)

Or

A [Converter Object](#)

value **(double)** - Input Fixed Frequency in Hertz.

Return Type Double

Default Not Applicable

Examples `mixer.InputFixedFrequency = 1e9`

C++ Syntax HRESULT get_InputFixedFrequency(double *pVal)

HRESULT put_InputFixedFrequency(double newVal)

Interface IMixer6

IConverter

Last Modified:

2-Feb-2009 Added converter

4-Mar-2008 Added note.

IsInputGreaterThanLO Property

Description Specifies whether to use the Input frequency that is greater than the LO or less than the LO. To learn more, see the [mixer setup dialog box help](#).

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

VB Syntax `obj.IsInputGreaterThanLO (LO) = bool`

Variable [\(Type\)](#) - [Description](#)

obj A [Mixer Interface](#) pointer to the [Measurement](#) (object)

Or

A [Converter Object](#)

LO (Integer) - LO stage number

Choose from 1 (default) or 2

bool **(Boolean)** -

True - Use the Input that is Greater than the specified LO.

False - Use the Input that is Less than the specified LO.

Return Type Boolean

Default True

Examples `mixer.IsInputGreaterThanLO(1) = True`

C++ Syntax HRESULT get_IsInputGreaterThanLO(VARIANT_BOOL * val);
HRESULT put_IsInputGreaterThanLO(VARIANT_BOOL val);

Interface IMixer2
IConverter

Last Modified:

2-Feb-2009 Added Converter

4-Mar-2008 Added note.

InputNumerator Property

Description Sets or returns the numerator value of the Input Fractional Multiplier.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

VB Syntax *obj*.**InputNumerator** = *value*

Variable [\(Type\)](#) - Description

obj A [Mixer Interface](#) pointer to the [Measurement](#) (object)

Or

A [Converter Object](#)

value (**Long**) - Input numerator value.

Return Type Long

Default 1

Examples

```
Print mixer.InputNumerator 'prints the value of the
InputNumerator
```

C++ Syntax HRESULT get_InputNumerator(long *pVal)

HRESULT put_InputNumerator(long newVal)

Interface IMixer

IConverter

Last Modified:

2-Feb-2009 Added converter

4-Mar-2008 Added note.

InputPower Property

Description Sets or returns the value of the Input Power.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

VB Syntax *obj*.**InputPower** = *value*

Variable [\(Type\)](#) - Description

obj A [Mixer Interface](#) pointer to the [Measurement](#) (object)

Or

A [Converter Object](#)

value **(double)** - Input power in dBm.

Return Type Double

Default -15 dBm for IMixer

-20 dBm for IConverter

Examples `Print mixer.InputPower 'prints the value of the InputPower`

C++ Syntax HRESULT get_InputPower(double *pVal)

HRESULT put_InputPower(double newVal)

Interface IMixer

IConverter

Last Modified:

2-Feb-2009 Added converter

4-Mar-2008 Added note.

InputRangeMode Property

Description Sets or returns the Input sweep mode.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

Note: There is also a [InputRangeMode Property](#) on the Converter Object.

VB Syntax *obj*.InputRangeMode = *value*

Variable [\(Type\)](#) - Description

obj A [Mixer Interface](#) pointer to the [Measurement](#) (object)

value **(Enum as MixerRangeMode)** - Input sweep mode. Choose from:

0 - **mixSwept**

1 - **mixFixed**

Return Type Enum

Default 0 - mixSwept

Examples `mixer.InputRangeMode = mixSwept`

C++ Syntax HRESULT get_InputRangeMode(long *pVal)

HRESULT put_InputRangeMode(long newVal)

Interface IMixer6

Last Modified:

4-Mar-2008 Added note.

InputRangeMode Property

Description Sets or returns the Input sweep mode.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

Note: There is also a [InputRangeMode](#) on the Mixer Interface.

VB Syntax *obj*.InputRangeMode = *value*

Variable [\(Type\)](#) - Description

obj A [Converter Object](#)

value **(Enum as NARangeMode)** - Input sweep mode. Choose from:

0 - **naSwept**

1 - **naFixed**

Return Type Enum

Default 0 - naSwept

Examples `conv.InputRangeMode = naSwept`

C++ Syntax HRESULT get_InputRangeMode(long LO, enum *pVal)

HRESULT put_InputRangeMode(long LO, enum newVal)

Interface IConverter

Last Modified:

18-Feb-2011 MX New topic

InputStartFrequency Property

Description Sets and returns the start frequency value of the mixer Input frequency.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

VB Syntax *obj*.InputStartFrequency = *value*

Variable [\(Type\)](#) - Description

obj A [Mixer Interface](#) pointer to the [Measurement](#) (object)

Or

A [Converter Object](#)

value (**double**) - Input start frequency in Hertz.

Return Type Double

Default Start frequency of the PNA

Examples `mixer.InputStartFrequency = Start_Freq`

C++ Syntax HRESULT get_InputStartFrequency(double *pVal)

HRESULT put_InputStartFrequency(double newVal)

Interface IMixer

IConverter

Last Modified:

2-Feb-2009 Added converter

4-Mar-2008 Added note.

InputStartPower Property

Description Sets and returns the Start Power value of the mixer Input Power.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

VB Syntax *obj*.**InputStartPower** = *value*

Variable [\(Type\)](#) - Description

obj A [Converter Object](#)

value **(double)** - Input start Power in dBm.

Return Type Double

Default Start Power of the PNA

Examples `mixer.InputStartPower = 0`

C++ Syntax HRESULT get_InputStartPower(double *pVal)

HRESULT put_InputStartPower(double newVal)

Interface IConverter

Last Modified:

2-Feb-2009 New topic

InputStopFrequency Property

Description Sets and returns the stop frequency value of the mixer Input frequency.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

VB Syntax *obj*.InputStopFrequency = *value*

Variable [\(Type\)](#) - Description

obj A [Mixer Interface](#) pointer to the [Measurement](#) (object)

Or

A [Converter Object](#)

value (**double**) - Input stop frequency in Hertz.

Return Type Double

Default Stop frequency of the PNA

Examples `mixer.InputStopFrequency = Stop_Freq`

C++ Syntax HRESULT get_InputStopFrequency(double *pVal)

HRESULT put_InputStopFrequency(double newVal)

Interface IMixer

IConverter

Last Modified:

2-Feb-2009 Added Converter

4-Mar-2008 Added note.

InputStopPower Property

Description Sets and returns the Stop Power value of the mixer Input Power.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

VB Syntax *obj*.**InputStopPower** = *value*

Variable [\(Type\)](#) - Description

obj A [Converter Object](#)

value **(double)** - Input Stop Power in dBm.

Return Type Double

Default Stop Power of the PNA

Examples `mixer.InputStopPower = 0`

C++ Syntax HRESULT get_InputStopPower(double *pVal)

HRESULT put_InputStopPower(double newVal)

Interface IConverter

Last Modified:

2-Feb-2009 New topic

InternalTestsetPortCount Property

Description	Returns the number of PNA test ports. This does not include the ports on an external test set .
VB Syntax	<i>value</i> = <i>cap</i> . InternalTestsetPortCount
Variable	(Type) - Description
<i>value</i>	(Long) - Variable to store the returned number of PNA test ports.
<i>cap</i>	A Capabilities (object)
Return Type	Long
Default	Not Applicable
Examples	<code>value = cap.InternalTestsetPortCount 'Read</code>
C++ Syntax	HRESULT get_InternalTestsetPortCount(long *numPorts);
Interface	ICapabilities

Interpolate Correction Property

Description Turns ON and OFF correction interpolation which calculates new error terms when stimulus values change after calibration.

When this property is ON and error correction is being applied, the calibration subsystem attempts to interpolate the error terms whenever the stimulus parameters are changed.

When this property is OFF under the same circumstances, error correction is turned OFF.

VB Syntax `meas.InterpolateCorrection = value`

Variable [\(Type\)](#) - Description

meas A [Measurement \(object\)](#)

value (**boolean**) - Choose from:

True - Turns correction interpolation ON

False - Turns correction interpolation OFF

Return Type Boolean

Default True

Examples

```
meas.InterpolateCorrection = False
calInterpolate = InterpolateCorrection 'Read
```

C++ Syntax HRESULT get_InterpolateCorrection(boolean *pVal)
HRESULT put_InterpolateCorrection(boolean newVal)

Interface IMeasurement

Interpolated Property

Description Turns marker Interpolation ON and OFF. Marker interpolation enables X-axis resolution beyond the discrete data values. The analyzer will calculate the x and y-axis data values between discrete data points. Use `meas.Interpolate` to change interpolation of **all** markers in a measurement. This command will override the measurement setting.

VB Syntax `mark.Interpolated = value`

Variable [\(Type\)](#) - Description

mark A [Marker](#) (object)

value (boolean)

False - Turns interpolation OFF

True - Turns interpolation ON

Return Type Boolean

Default True

Examples

```
mark.Interpolated = True 'Write
```

```
interpolate = mark.Interpolated 'Read
```

C++ Syntax HRESULT get_Interpolated(VARIANT_BOOL *pVal)
HRESULT put_Interpolated(VARIANT_BOOL newVal)

Interface IMarker

InterpolateNormalization Property **Superseded**

Description **Note:** This property is replaced by [DoReceiverPowerCal Method](#).

Turns ON and OFF normalization interpolation which calculates new divisor data when stimulus values change after normalization.

When this property is ON and normalization is being applied, the Normalization algorithm attempts to interpolate the divisor data whenever the stimulus parameters are changed.

When this property is OFF under the same circumstances, normalization is turned OFF.

Normalization is currently supported only on measurements of unratiod power for the purpose of performing a receiver power calibration.

VB Syntax *meas*.InterpolateNormalization = *value*

Variable [\(Type\)](#) - Description

meas **(object)** - A Measurement object

value **(boolean)**

False – Turns normalization interpolation OFF

True – Turns normalization interpolation ON

Return Type Boolean

Default False -OFF

Examples `meas.InterpolateNormalization = False` 'Write

`normalized = meas.InterpolateNormalization` 'Read

C++ Syntax HRESULT put_InterpolateNormalization(VARIANT_BOOL bState);

HRESULT get_InterpolateNormalization(VARIANT_BOOL *bState);

Interface IMeasurement

Interrupt Property

Description	Reads the boolean that represents the state of the Interrupt In line (pin 13) on the external test set connector.
VB Syntax	<i>value</i> = <i>ExtIO</i> . Interrupt
Variable	(Type) - Description
<i>value</i>	(boolean) - Variable to store the returned data
<i>ExtIO</i>	(object) - An ExternalTestSetIO object
Return Type	Boolean
	False - indicates the line is being held at a TTL High
	True - indicates the line is being held at a TTL Low
Default	Not Applicable
Examples	<code>value = ExtIO.Interrupt</code>
C++ Syntax	HRESULT get_Interrupt(VARIANT_BOOL* bValue);
Interface	IHWExternalTestSetIO

Invert Property

Description Sets whether to invert the polarity of the pulse.

VB Syntax *pulse*.**Invert** (*n*) = *value*

Variable [\(Type\)](#) - Description

pulse A [PulseGenerator](#) (**object**)

n (**Integer**) Pulse generator number. Choose from 0 to 4.

Or use [PulseGeneratorID](#) to refer to an external pulse generator.

0 is the generator that pulses the ADC.

value **True** - Invert the pulse generator polarity. This causes the pulse ON time to be active low and OFF be active high.

False - Do NOT Invert the pulse generator polarity.

Return Type Boolean

Default False

Examples

```
pulse.Invert(1) = True 'Write
```

```
value = pulse.Invert(4) 'Read
```

C++ Syntax

```
HRESULT get_Invert(VARIANT_BOOL *pVal);
```

```
HRESULT put_Invert(VARIANT_BOOL newVal);
```

Interface IPulseGenerator3

Last Modified:

16-Feb-2012 Edit for Ext PG

8-Oct-2010 MX New topic

IOConfiguration Property

Description Sets and returns the method of communication and address for the external device.

VB Syntax *extDevices.IOConfiguration = value*

Variable [\(Type\)](#) - Description

extDevices An [ExternalDevice](#) (object)

value (String) Configuration path. Any valid VISA resource shown in the IO Configuration field of the [External Devices dialog](#), enclosed in quotes.

Do NOT use the ID string of a PMAR USB power sensor as the resource string. The ID string is returned by the [USBPowerMeterCatalog Property](#)

Return Type String

Default " " Empty String

Examples

```
extDevices.IOConfiguration =  
"TCPIP::141.121.76.239::inst0::INSTR" Write  
  
value = extDevices.IOConfiguration 'Read  
  
See example program to configure PMAR device  
  
See example program to configure External Source
```

C++ Syntax HRESULT get_IOConfiguration(BSTR* value);
HRESULT put_IOConfiguration(BSTR newVal);

Interface IExternalDevices

Last Modified:

31-Jul-2009 MX New topic

IOEnable Property

Description Sets and returns whether an external device is enabled for IO communication with the PNA.

When disabled (False), the PNA will NOT attempt to connect to the external device regardless of the instrument state ([Active](#)).property. Therefore, no errors will be produced if the device is not connected.

This command is useful for debugging and testing states when the external device is not connected. This command is unnecessary in ordinary operation (when the device is connected).

VB Syntax `extDevices.IOEnable = value`

Variable [\(Type\)](#) - Description

extDevices An [ExternalDevice](#) (object)

value (Boolean) Choose from:

True - Device is available for IO communication.

False - Device is NOT available for IO communication.

Return Type Boolean

Default True

Examples

```
extDevices.IOEnable = True 'Write
```

```
bool = extDevices.IOEnable 'Read
```

See example program to configure [PMAR device](#)

See example program to configure [External Source](#)

C++ Syntax HRESULT get_IOEnable(VARIANT_BOOL* value);

HRESULT put_IOEnable(VARIANT_BOOL newVal);

Interface IExternalDevices

Last Modified:

17-Sep-2012 Edited description

31-Jul-2009 MX New topic

IsContinuous Property

Description	Returns whether or not a channel is in continuous mode. To set the channel to continuous mode, use Continuous Method .
VB Syntax	<i>value</i> = <i>chan</i> . IsContinuous
Variable	(Type) - Description
<i>value</i>	(boolean) - Choose either: False - Channel trigger is NOT set to continuous. True - Channel trigger IS set to continuous.
<i>chan</i>	Channel (object)
Return Type	Boolean
Default	Not Applicable
Examples	<pre>trig = chan.IsContinuous 'Read</pre>
C++ Syntax	HRESULT get_IsContinuous (VARIANT_BOOL *bContinuous);
Interface	IChannel4

Last Modified:

13-Nov-2013 Fixed Channel rev.

IsDevicePresent Property

Description	Returns whether the named device is present on the bus for which it is configured.
VB Syntax	<i>value</i> = <i>extDevices</i> . IsDevicePresent (<i>devName</i>)
Variable	(Type) - Description
<i>value</i>	(Boolean) Variable to store the returned value: <ul style="list-style-type: none">• False - The device is not in the collection or the device fails to respond and times out when communication is attempted.• True - The device responds when communication is attempted.
<i>extDevices</i>	An ExternalDevices (collection)
<i>devName</i>	(String) Name of the device with which you want to communicate.
Return Type	Boolean
Default	Not Applicable
Examples	<pre>present = extDevices.IsDevicePresent("MyPowerMeter") 'Read</pre>
C++ Syntax	HRESULT get_IsDevicePresent(VARIANT_BOOL *present);
Interface	IExternalDevices2

Last Modified:

19-Jul-2011 New topic

IsECALModuleFoundEx Property

Description	This property replaces IsECALModuleFound Property . Returns true or false depending on whether communication was established between the PNA and the specified ECal module.
VB Syntax	<i>modFound</i> = <i>cal</i> . IsECALModuleFoundEx (<i>module</i>)
Variable	(Type) - Description
<i>modFound</i>	(boolean) - Variable to store the returned test result. True - The PNA identified the presence of the specified ECal module. False - The PNA did NOT identify the presence of the specified ECal module.
<i>cal</i>	(object) - A Calibrator object
<i>module</i>	(long integer) ECal module. Choose from modules 1 through 8 Use GetECALModuleInfoEx to return the model and serial number of each module.
Return Type	Boolean
Default	Not applicable
Examples	<pre>Set cal = pna.ActiveChannel.Calibrator moduleFound = cal.IsECALModuleFoundEx(1)</pre>
C++ Syntax	HRESULT get_IsECALModuleFoundEx(long moduleNumber, VARIANT_BOOL *bModuleFound);
Interface	ICalibrator4

IsFrequencyOffsetPresent Property

Description	Returns a value indicating the presence of Frequency Offset Option 080 in the remote PNA.
VB Syntax	<i>value</i> = <i>cap</i> . IsFrequencyOffsetPresent
Variable	(Type) - Description
<i>value</i>	(Boolean) - Variable to store the returned value True - Frequency Offset Option 080 is present False - Frequency Offset Option 080 is not present
<i>cap</i>	A Capabilities (object)
Return Type	Boolean
Default	Not Applicable
Examples	<pre>value = cap.isFrequencyOffsetPresent(1) 'Read</pre>
C++ Syntax	HRESULT get_IsFrequencyOffsetPresent(VARIANT_BOOL * present);
Interface	ICapabilities

IsHold Property

Description	Returns whether or not a channel is in hold mode. To set the channel to hold mode, use Hold Method .
VB Syntax	<i>value</i> = <i>chan</i> . IsHold
Variable	(Type) - Description
<i>value</i>	(boolean) - Choose either: False - Channel trigger is NOT set to hold. True - Channel trigger IS set to hold.
<i>chan</i>	Channel (object)
Return Type	Boolean
Default	Not Applicable
Examples	<pre>trig = chan.IsHold 'Read</pre>
C++ Syntax	HRESULT get_IsHold (VARIANT_BOOL *bHold);
Interface	IChannel3

IsMarkerOn Property

Description Returns whether or not a marker was used for the specified tuning sweep.

VB Syntax *value* = *embedLODiag*.**IsMarkerOn** (*n*)

Variable [\(Type\)](#) - [Description](#)

value **(Boolean)** Variable to store the returned data.

embedLODiag An [EmbeddedLODiagnostic](#) (**object**)

n Tuning sweep number. Use [NumberOfSweeps](#) to find the number of sweeps taken.

Return Type **(String)**

Default Not Applicable

Examples `data= embedLO.IsMarkerOn 3 'read`

C++ Syntax HRESULT IsMarkerOn(long sweep, VARIANT_BOOL* markerOn);

Interface IEmbeddedLODiagnostic

Last Modified:

13-Apr-2007 MX New topic

IsolationIncrementAveraging Property

Description Value by which to increment (increase) the channel's averaging factor during measurement of isolation in an ECal calibration.

Note: If <value> is greater than 1 and the channel currently has averaging turned OFF, averaging will be turned ON only during the isolation measurements and with the averaging factor equal to <value>.

VB Syntax `cal.IsolationIncrementAveraging = value`

Variable [\(Type\) - Description](#)

cal A [Calibrator](#) (object)

value **(Long)** Incremental Averaging factor. The maximum averaging factor is 65536 (2¹⁶).

Return Type Long Integer

Default 8

Examples

```
oCal.IsolationAveragingIncrement = 16 'Write  
avgIncr = oCal.IsolationAveragingIncrement ' Read
```

C++ Syntax

```
HRESULT get_IsolationAveragingIncrement(long *pVal);  
HRESULT put_IsolationAveragingIncrement(long newVal);
```

Interface ICalibrator7

Last Modified:

16-Apr-2007 MX New topic

IsOn Property

Description Sets and returns the ON |OFF state of Embedded LO.

VB Syntax *obj.IsOn = value*

Variable [\(Type\)](#) - Description

obj An [EmbeddedLO](#) (**object**) or

A [ConverterEmbeddedLO](#) (**object**)

value (**Boolean**)

False - Turns Embedded LO OFF

True - Turns Embedded LO ON

Return Type (**Boolean**)

Default False (OFF)

Examples

```
embedLO.IsOn = True 'write
```

```
data= embedLO.IsOn 'read
```

C++ Syntax HRESULT get_IsOn(VARIANT_BOOL* IsOn);

HRESULT put_IsOn(VARIANT_BOOL IsOn);

Interface IEmbeddedLO

Last Modified:

12-Aug-2009 Added ConvEmbedLO object

18-Apr-2007 MX New topic

IsReceiverStepAttenuatorPresent Property

Description	Returns a value indicating the presence of Receiver step attenuators in the remote PNA.
VB Syntax	<i>value</i> = <i>cap</i> . IsReceiverStepAttenuatorPresent (<i>n</i>)
Variable	(Type) - Description
<i>value</i>	(Boolean) - Variable to store the returned value. True - Receiver step attenuators are present. False - Receiver step attenuators are not present.
<i>cap</i>	A Capabilities (object)
<i>n</i>	(Long) - port number to query for step attenuators
Return Type	Boolean
Default	Not Applicable
Examples	<pre>value = cap.IsReceiverStepAttenuatorPresent(1) 'Read</pre>
C++ Syntax	HRESULT get_IsReceiverStepAttenuatorPresent(long portNumber, VARIANT_BOOL * present);
Interface	ICapabilities

IsReferenceBypassSwitchPresent Property

Description	Returns a value indicating the presence of a Reference Bypass Switch in the remote PNA.
VB Syntax	<i>value</i> = <i>cap</i> .IsReferenceBypassSwitchPresent (<i>n</i>)
Variable	(Type) - Description
<i>value</i>	(Boolean) - Variable to store the returned value. True - Reference Bypass Switch is present. False - Reference Bypass Switch is not present.
<i>cap</i>	A Capabilities (object)
<i>n</i>	(Long) - port number to query for reference bypass switch
Return Type	Boolean
Default	Not Applicable
Examples	<pre>value = cap.IsReferenceBypassSwitchPresent(1) 'Read</pre>
C++ Syntax	HRESULT get_IsReferenceBypassSwitchPresent(long portNumber, VARIANT_BOOL * present);
Interface	ICapabilities

Read-only

IsSParameter Property

Description	Returns true if measurement represents an S-Parameter
VB Syntax	<i>value</i> = <i>meas</i> . IsSparameter
Variable	(Type) - Description
<i>meas</i>	A Measurement (object)
<i>value</i>	(Boolean)
	True - measurement is an S-Parameter
	False - measurement is NOT an S-Parameter
Return Type	Boolean
Default	True
Examples	<pre>print app.IsSparameter</pre>
C++ Syntax	HRESULT IsSparameter([out, retval] VARIANT_BOOL * bVal);
Interface	IMeasurement2

Item Property

Description Add or change a name-value pair in the Cal Set, or read the value associated with a name.

After editing, [Save](#) the CalSet to the PNA.

About Name-Value pairs

A Cal Set name-value pair is a general purpose data structure that maps a name to a value. This property allows you to associate a name with a value. Then, using this same property, you can read the value using the name.

For example, one of the items added by the PNA firmware to every Cal Set is named 'Created By'. The value attached to this item is the name of the PNA App that created the Cal Set. When an SMC cal is performed, you can query the Cal Set for the 'Create By' item, and it will return 'Scalar Mixer/Converter'. The same query on an NFX channel returns 'Noise Figure Converters'.

Warning - Do NOT change the name or value of any Items that you did NOT create. Otherwise, the PNA firmware may behave unpredictably.

See Also

[EnumerateItems Method](#)

[RemoveItem Method](#)

VB Syntax *CalSet.Item (name) = value*

Variable [\(Type\)](#) - Description

CalSet **(object)** - A [Cal Set](#) object

name **(String)** - Name of the name-value pair.

value **(Variant)** - Can be an integer, float, double, string, or a single-dimensioned array of integer, float, double, string.

Return Type Variant

Default Not Applicable

Example

```
' Create the pna object
Dim pna
Set pna = CreateObject("AgilentPNA835x.Application")
```

```

' Get a handle to the calsets collection
Dim calsets
Set calsets = pna.GetCalManager.calsets
' Get a handle to the cal set to be edited
Dim MyCalSet
Set MyCalSet = calsets.Item("CalSet_1")
' Add a name-value pair(item) to MyCalSet
MyCalSet.Item("MyItem")=15
' Save the edited Cal Set to the PNA
MyCalSet.Save
' Loop thru the name-value pairs in the Cal Set
Dim CSetItems
CSetItems = MyCalSet.EnumerateItems
for i=lbound(CSetItems) to Ubound(CSetItems)
    ' List the item names in MyCalSet
    Dim name
    name = CSetItems(i)
    wscript.echo name
    ' List the value for each item name
    Dim value
    value = myCalSet.Item(name)
    wscript.echo value
Next
' Delete the new name-value pair
MyCalSet.RemoveItem("MyItem")

```

[C++ Syntax](#) HRESULT get_Item(BSTR name, VARIANT *value);
 HRESULT put_Item(BSTR name, VARIANT value);

Interface ICalSet6

Last Modified:

24-Sep-2010 MX New topic

Items Property

Description	Returns a list of configured external devices in the collection.
VB Syntax	<code>array = extDevices.Items</code>
Variable	(Type) - Description
<i>value</i>	(array) Variable to store the returned values. Each item in the array is in the format (name:driver), where: <ul style="list-style-type: none">• name = name of the external device• driver = See a list of supported drivers.
<i>extDevices</i>	An ExternalDevices (collection)
Return Type	Variant Array
Default	Not Applicable
Examples	<code>array = extDevices.Items 'Read</code>
C++ Syntax	<code>HRESULT get_Items(VARIANT* array);</code>
Interface	IExternalDevices

Last Modified:

31-Jul-2009 MX New topic

IterationNumber Property

Description	Sets and returns the maximum leveling sweep iterations to be used in order to achieve the tolerance setting.
VB Syntax	<i>RxLevel</i> . IterationNumber (<i>srcPort</i>) = <i>value</i>
Variable	(Type) - Description
<i>RxLevel</i>	A ReceiverLeveling Object
<i>srcPort</i>	(Long Integer) Source port for which to set the IterationNumber for Receiver Leveling.
	Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use chan.getPortNumber to translate the string into a port number. To learn more see Remotely Specifying a Source Port
<i>value</i>	(Long Integer) Max iterations. Choose a value between 1 and 25.
Return Type	(Long Integer)
Default	5
Examples	<pre>rxLevel.IterationNumber (1) = 10 ' Write value = rxLevel.IterationNumber 2 ' Read</pre>
C++ Syntax	<pre>HRESULT get_IterationNumber(long port, long* pVal); HRESULT put_IterationNumber(long port, long newVal);</pre>
Interface	IRceiverLevelingConfiguration

Last Modified:

13-Feb-2009 MX New topic

IterationsTolerance Property

Description This command, along with [MaximumIterationsPerPoint Property](#) deal with ADJUSTMENTS made to the source power.

Sets the maximum desired deviation from the sum of the [test port power](#) and the [offset](#) value. Power readings will continue to be made, and source power adjusted, until a reading is within this tolerance value or the [max number of readings](#) has been met. The last value to be read is the valid reading for that data point.

The following two commands allow for settling of power READINGS.

[ReadingsPerPoint Property](#)

[ReadingsTolerance Property](#)

VB Syntax *pwrCal.IterationsTolerance = value*

Variable [\(Type\)](#) - Description

pwrCal **(object)** - A [SourcePowerCalibrator](#) (object)

value **(Double)** – Tolerance value in dBm. Choose any number between 0 and 5

Return Type Double

Default .05 dB

Examples

```
Set powerCalibrator = pna.SourcePowerCalibrator
powerCalibrator.IterationsTolerance = .1 'Write
ReadTol = powerCalibrator.IterationsTolerance 'Read
```

C++ Syntax HRESULT get_IterationsTolerance(double *pVal);

HRESULT put_IterationsTolerance(double newVal);

Interface ISourcePowerCalibrator3

Last Modified:

17-Apr-2007 Clarified verbage

KaiserBeta Property

Description Sets or returns the Kaiser Beta of Time Domain transform windows

VB Syntax *trans*.KaiserBeta = *value*

Variable [\(Type\)](#) - Description

trans A Transform (**object**)

value (**single**) - Kaiser Beta. Choose any number between **0** and **13**.

Return Type Single

Default 0

Examples `trans.KaiserBeta = 6 'sets the Kaiser Beta of a transform window`
`-Write`

`KB = trans.KaiserBeta 'Read`

C++ Syntax HRESULT get_KaiserBeta(float *pVal)
HRESULT put_KaiserBeta(float newVal)

Interface ITransform

L0 Property

Description Sets and Returns the L0 (L-zero) value (the first inductance value) for the calibration standard.

To set the other inductance values, use [L1](#), [L2](#), [L3](#).

VB Syntax `calstd.L0 = value`

Variable [\(Type\)](#) - [Description](#)

calstd A CalStandard (**object**). Use calKit.[GetCalStandard](#) to get a handle to the standard.

value (**single**) - Value for L0 in femtohenries(1E-15)

Return Type Single

Default Not Applicable

Examples `calstd.L0 = 15 'Write the value of L0 = 15 femtohenries`

`Induct0 = calstd.L0 'Read the value of L0`

C++ Syntax HRESULT get_L0(float *pVal)
HRESULT put_L0(float newVal)

Interface ICalStandard

L1 Property

Description Sets and Returns the L1 value (the second inductance value) for the calibration standard.

To set the other inductance values, use [L0](#), [L2](#), [L3](#).

VB Syntax `calstd.L1 = value`

Variable [\(Type\)](#) - Description

calstd A CalStandard (**object**). Use calKit.[GetCalStandard](#) to get a handle to the standard.

value (**single**) - Value for L1.

Return Type Single

Default Not Applicable

Examples

```
calstd.L1 = 15 'Write the value of L1
```

```
Induct1 = calstd.L1 'Read the value of L1
```

C++ Syntax HRESULT get_L1(float *pVal)
HRESULT put_L1(float newVal)

Interface ICalStandard

L2 Property

Description Sets and Returns the L2 value (the third inductance value) for the calibration standard.

To set the other inductance values, use [L0](#), [L1](#), [L3](#).

VB Syntax `calstd.L2 = value`

Variable [\(Type\)](#) - Description

calstd A CalStandard (**object**). Use calKit.[GetCalStandard](#) to get a handle to the standard.

value (**single**) - Value for L2.

Return Type Single

Default Not Applicable

Examples

```
calstd.L2 = 15 'Write the value of L2.
```

```
Induct2 = calstd.L2 'Read the value of L2
```

C++ Syntax HRESULT get_L2(float *pVal)
HRESULT put_L2(float newVal)

Interface ICalStandard

L3 Property

Description Sets and Returns the L3 value (the third inductance value) for the calibration standard.

To set the other inductance values, use [L0](#), [L1](#), [L2](#).

VB Syntax `calstd.L3 = value`

Variable [\(Type\)](#) - Description

calstd A CalStandard (**object**). Use calKit.[GetCalStandard](#) to get a handle to the standard.

value (**single**) - Value for L3.

Return Type Single

Default Not Applicable

Examples `calstd.L3 = 15 'Write the value of L3.`

`Induct3 = calstd.L3 'Read the value of L3`

C++ Syntax HRESULT get_L3(float *pVal)
HRESULT put_L3(float newVal)

Interface ICalStandard

Label Property

Description Sets and Returns the label for the calibration standard. The label is used to prompt the user to connect the specified standard.

VB Syntax *calstd.Label* = *value*

Variable [\(Type\)](#) - Description

calstd A CalStandard (**object**). Use calKit.[GetCalStandard](#) to get a handle to the standard.

value (**string**) - between 1 and 12 characters long. Cannot begin with a numeric.

Return Type String

Default Not Applicable

Examples `calstd.Label = "Short" 'Write`

`stdLabel = calstd.Label 'Read`

C++ Syntax HRESULT get_Label(BSTR *pVal)
HRESULT put_Label(BSTR newVal)

Interface ICalStandard

Label Property

Description	Sets or gets the display label for a given channel/testset combination. The label appears in a status bar at the bottom of the PNA display when the ShowProperties property is set to TRUE.
VB Syntax	<i>tset</i> . Label (<i>chNum</i>) = <i>value</i>
Variable	(Type) - Description
<i>tset</i>	A TestsetControl object. Obtained from the ExternalTestsets collection.
<i>chNum</i>	(Integer) Channel number of the measurement.
<i>value</i>	(String) The text of the label.
Return Type	String
Default	None
Examples	<pre>'The following sets the label for channel 5 corresponding to a given testset object.'</pre> <pre>testset1.Label(5) = 'High-power output'</pre> See External Testset Program
C++ Syntax	HRESULT get_Label(long channelNum, BSTR *pLabel); HRESULT put_Label(long channelNum, BSTR label);
Interface	ITestsetControl

LANConfiguration Property

Description	Returns information about the current status of the PNA's computer networking configuration. This is the same set of information that is returned in an NA_IPConfiguration data structure by the GetIPConfigurationStruct method.
VB Syntax	<i>value</i> = <i>app</i> .LANConfiguration
Variable	(Type) - Description
<i>value</i>	(String) Variable to contain the PNA's LAN configuration string
<i>app</i>	An Application (object)
Return Type	Comma-delimited string
Default	Not Applicable
Examples	<pre>networkConfigInfo = app.LANConfiguration</pre>
C++ Syntax	HRESULT get_LANConfiguration (BSTR * pStrConfig);
Interface	IApplication13

Last Modified:

2-Jun-2008 MX New topic

LastCalPassedTolerance Property

Description	Returns the pass / fail status of the tolerance limits of the target power from the most recent source power cal.
VB Syntax	<i>value</i> = <i>pwrCal</i> . LastCalPassedTolerance
Variable	(Type) - Description
<i>pwrCal</i>	(object) – A SourcePowerCalibrator (object)
<i>value</i>	(boolean) - False – Source power cal did NOT achieve the specified tolerance limits. True – Source power cal DID achieve the specified tolerance limits.
Return Type	Boolean
Default	Not Applicable
Examples	<pre>status = powerCal.LastCalPassedTolerance 'Read</pre>
C++ Syntax	HRESULT get_LastCalPassedTolerance(VARIANT_BOOL *bState);
Interface	ISourcePowerCalibrator7

Last Modified:

17-Mar-2010 MX New topic

LastLevelingAsSPC Property

Description Sets and returns the state of **Use Last Result for Source Power Cal**. When Leveling Mode is switched back to Internal, this feature turns Source Power Cal correction ON using the latest receiver leveling correction data.

VB Syntax *RxLevel.LastLevelingAsSPC(srcPort) = value*

Variable [\(Type\)](#) - **Description**

RxLevel A [ReceiverLeveling](#) Object

srcPort (Long Integer) Source port being used for Receiver Leveling.

Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#)

value (Boolean) . Choose from:

True - When Leveling Mode is switched back to Internal, Source Power Cal correction is turned ON using the latest receiver leveling correction data.

False - When Leveling Mode is switched back to Internal, Source Power Cal correction is NOT turned ON.

Return Type Variant Boolean

Default False

Examples

```
rxLevel.LastLevelingAsSPC (1) = True ' Write
value = rxLevel.LastLevelingAsSPC 2 ' Read
```

C++ Syntax

```
HRESULT get_LastLevelingAsSPC(long port, VARIANT_BOOL* pVal);
HRESULT put_LastLevelingAsSPC(long port, VARIANT_BOOL newVal);
```

Interface IReceiverLevelingConfiguration2

Last Modified:

16-Sep-2010 MX New topic

LastModified Property

Description Returns the time stamp of the last modification to this Cal Set.

The time is returned in the local time zone setting of the PNA.

VB Syntax *value* = *Object*.**LastModified**

Variable [\(Type\)](#) - Description

object Channel (**object**)

or

CalSet (**object**) - Read-only property

value (**Variant**) – Variable to store the time stamp.

Return Type Variant

Default Not Applicable

Examples `date = CalSet.LastModified 'Read`

C++ Syntax HRESULT get_LastModified(VARIANT* datetime)

Interface ICalSet3

Last Modified:

13-May-2013 Removed GMT note
25-Apr-2007 Added GMT note

LevelingIFBW Property

Description Sets and returns the IFBW to be used for leveling sweeps. Enable separate IFBW for leveling sweeps using [FastMode Property](#).

VB Syntax *RxLevel.LevelingIFBW(srcPort) = value*

Variable [\(Type\)](#) - [Description](#)

RxLevel A [ReceiverLeveling](#) Object

srcPort (Long Integer) Source port for which to set the LevelingIFBW for Receiver Leveling.

Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#)

value (Double) IFBW for leveling sweeps in Hz. The list of valid IF Bandwidths is different depending on the PNA model. [See list](#). If an invalid number is specified, the PNA will round up to the closest valid number.

Return Type (Double)

Default 100 kHz

Examples

```
rxLevel.LevelingIFBW (1) = 1e3 ' Write
value = rxLevel.LevelingIFBW 2 ' Read
```

C++ Syntax

```
HRESULT get_LevelingIFBW(long port, double* pVal);
HRESULT put_LevelingIFBW(long port, double newVal);
```

Interface IReceiverLevelingConfiguration

Last Modified:

13-Feb-2009 MX New topic

Limit Property

Description Sets and returns the power limit for the specified port.

VB Syntax *gpl.Limit (port) = value*

Variable [\(Type\)](#) - Description

gpl A [GlobalPowerLimit](#) (**object**)

port (**Long**) Port number for which power limit value is to be set.

value (**Double**) Power limit value. Choose a value between -27 dBm (approximately) and the max settable power for the PNA.

Return Type Double

Default 100 dBm for all ports

Examples `gpl.Limit(1) = 0 'Write`

`Limit = gpl.Limit(2) 'Read`

C++ Syntax HRESULT get_Limit(long port, double *pVal)
HRESULT put_Limit(long port, double newVal)

Interface IGlobalPowerLimit

Last Modified:

10-Aug-2009 MX New topic

LimitFrequency Property

Description	Enable or disable the use of the power meter min and max frequencies.
VB Syntax	<i>pwrSensor.LimitFrequency = value</i>
Variable	(Type) - Description
<i>pwrSensor</i>	A PowerSensorAsReceiver (Object)
<i>value</i>	(Boolean) - State of min and max frequency use. Choose from: False - Min and max frequencies disabled. True - Min and max frequencies enabled.
Return Type	Boolean
Default	False
Examples	<pre>pwrSensor.LimitFrequency = False 'Write</pre> <pre>limit = pwrSensor.LimitFrequency 'Read</pre>
C++ Syntax	HRESULT put_LimitFrequency(VARIANT_BOOL newVal); HRESULT get_LimitFrequency(VARIANT_BOOL* pVal);
Interface	IPowerSensorAsReceiver

Last Modified:

25-Aug-9 MX New topic

LimitTestFailed Property

Description Returns the results of limit testing for the measurement.

VB Syntax *testFailed* = *meas*.**LimitTestFailed**

Variable [\(Type\)](#) - [Description](#)

testFailed (**boolean**) Variable to store the returned value

False - Limit Test Passed

True - Limit Test Failed

meas A Measurement (**object**)

Return Type Boolean

Default False returned if there is no testing in progress.

Examples

```
Dim testRes As Boolean
testRes = meas.LimitTestFailed
MsgBox (testRes)
```

C++ Syntax HRESULT get_LimitTestFailed(VARIANT_BOOL* trueIfFailed)

Interface IMeasurement

EndStimulus Property

Description When constructing a limit line, specifies the stimulus value for the end of the segment.

VB Syntax *limtseg*.EndStimulus = *value*

Variable [\(Type\)](#) - Description

limtseg A LimitSegment (**object**)

value (**double**) - End Stimulus X-axis value. No units

Return Type Double

Default 0

Examples

```
Set limtseg = meas.LimitTest(1)
limtseg.EndStimulus = 8e9 'Write
```

```
EndStim = limtseg.EndStimulus 'Read
```

C++ Syntax HRESULT get_EndStimulus(double *pVal)
HRESULT put_EndStimulus(double newVal)

Interface ILimitSegment

EndResponse Property

Description When constructing a limit line, specifies the amplitude value at the end of the limit segment.

VB Syntax `limtseg.EndResponse = value`

Variable [\(Type\)](#) - Description

limts A LimitSegment (**object**)

value (**double**) - Y-axis value of the End Response limit. No units

Return Type Double

Default 0

Examples

```
Set limtseg = meas.LimitTest(1)
limtseg.EndResponse = 10 'Write
EndResp = limtseg.EndResponse 'Read
```

C++ Syntax HRESULT get_EndResponse(double *pVal)
HRESULT put_EndResponse(double newVal)

Interface ILimitSegment

Type (limit) Property

Description Specifies the Limit Line type.

VB Syntax `limit(index).Type = value`

Variable [\(Type\)](#) - Description

limit A LimitSegment (**object**)

index (**variant**) - Limit line number in the LimitTest collection

value (**enum NALimitSegmentType**) - Limit Line type. Choose from:

0 - naLimitSegmentType_OFF - turns limit line OFF

1 - naLimitSegmentType_Maximum - limit line fails with a data point ABOVE the line

2 - naLimitSegmentType_Minimum - limit line fails with a data point BELOW the line

Return Type Long Integer

Default 0 - OFF

Examples

```
Set limits = meas.LimitTest
limits.Type = naLimitSegmentType_Maximum 'Write
limitType = limits.Type 'Read
```

C++ Syntax HRESULT put_Type(tagNALimitSegmentType *pVal)
HRESULT get_Type(tagNALimitSegmentType newVal)

Interface ILimitSegment

LineDisplay Property

Description Turns the display of limit lines ON or OFF. To turn limit TESTING On and OFF, use [State Property](#).

Note: Trace data must be ON to view limit lines

VB Syntax *limitst.LineDisplay = state*

Variable [\(Type\)](#) - Description

limitst A LimitTest (**object**)

state (**boolean**)

False - Turns the display of limit lines OFF

True - Turns the display of limit lines ON

Return Type Long Integer

Default **True**

Examples `Limtttest.LineDisplay = true 'Write`

`lineDsp = Limtttest.LineDisplay 'Read`

C++ Syntax HRESULT get_LineDisplay(VARIANT_BOOL *pVal)
HRESULT put_LineDisplay(VARIANT_BOOL newVal)

Interface ILimitTest

ListData Property

Description Sets and returns the DC stimulus values per point from the specified DC source.

This setting overrides the Start and Stop DC settings for the channel. Only the values that are set with this command can be read by this command. The read command does NOT read the values that are set using the Start and Stop settings.

VB Syntax *DCStim.ListData(name,port) = values*

Variable [\(Type\)](#) - Description

DCStim A [DCStimulus](#) (object)

name,port (String) Name of the DC source and source port.

Use Source Property to read a list of configured DC source names.

To set the DC source to be always ON, do NOT specify a port.

value (Variant) DC stimulus value array. The size of the array must equal the sweep point number.

Return Type Variant array

Default Not Applicable

Examples

```
'The following shows how to set a DC value for each data point
dim data(2)
data(0) = 1
data(1) = 2
data(2) = 1
DC.ListData("A01") = data
'Read
varData = dc.ListData("A01")
for i=0 to 2
msgbox (varData (i))
next
```

C++ Syntax HRESULT get_ListData(BSTR name, VARIANT * pValue);
HRESULT put_ListData(BSTR name, VARIANT newValue);

Interface IExternalDCSource

Last modified:

14-Nov-2012 New topic

LoadCharFromFile Property

Description	Sets and returns whether a Mixer characterization file is to be loaded. Specify and load the filename with CharFileName Property
VB Syntax	<i>VMC.LoadCharFromFile = bool</i>
Variable	(Type) - Description
<i>VMC</i>	VMCType (object)
<i>bool</i>	(Boolean)
	True - Load from file
	False - Perform mixer characterization
Return Type	Boolean
Default	False
Examples	<code>value = VMC.LoadCharFromFile</code>
C++ Syntax	HRESULT put_LoadCharFromFile(VARIANT_BOOL bLoadCharFromFile); HRESULT get_LoadCharFromFile(VARIANT_BOOL *bLoadCharFromFile);
Interface	VMCType

LoadImpedance Property

Description Sets and returns the load impedance of the pulse generator.

VB Syntax *extPulseGen*.**LoadImpedance** = *value*

Variable [\(Type\)](#) - [Description](#)

extPulseGen An [ExternalPulseGenerator](#) (**object**)

value (**Double**) Pulse generator load impedance.

Return Type Double

Default **50**

Examples `extPulseGen.LoadImpedance = 50` 'Write

`load = extPulseGen.LoadImpedance` 'Read

C++ Syntax HRESULT get_LoadImpedance (double *pValue)

HRESULT put_LoadImpedance (double newVal)

Interface IExternalPulseGenerator

Last Modified:

15-Feb-2012 New topic

LoadPort Property

Description Returns the load port number associated with an S-parameter reflection measurement. If the measurement is not a reflection S-parameter, the number returned by this property will have no meaning.

VB Syntax `loadPort = meas.LoadPort`

Variable [\(Type\)](#) - **Description**

`loadPort` **(long integer)** - The reflection measurement's load port number.

`meas` A Measurement **(object)**

Return Type Long Integer

Default Not Applicable

Examples

```
Set meas = pna.ActiveMeasurement
loadPort = meas.LoadPort
```

C++ Syntax HRESULT get_LoadPort(long *pPortNumber);

Interface IMeasurement

LocalLockoutState Property

Description	Prevents use of the mouse, keyboard, and front panel while your program is running. Use of these controls while this property is set TRUE causes an error message on the PNA display. To prevent these messages, see About Error Messages .
VB Syntax	<i>app.LocalLockoutState = bool</i>
Variable	(Type) - Description
<i>app</i>	An Application (object)
<i>bool</i>	(boolean) -Choose either: False - User Interface is NOT locked out. True - User Interface IS locked out.
Return Type	Boolean
Default	False
Examples	<pre>app.LocalLockoutState = True 'Write block = app.LocalLockoutState 'Read</pre>
C++ Syntax	HRESULT get_LocalLockoutState(VARIANT_BOOL *State); HRESULT put_LocalLockoutState(VARIANT_BOOL *State);
Interface	IApplication4

Locator Property

Description Specifies the location, address, or ID string of the power meter / sensor that is used during a source power calibration.

Use [Path Property](#) to specify the type of interface.

VB Syntax `pwrMtrInterface.Locator = value`

Variable (Type) - Description

`pwrMtrInterface` **(object)** - A [PowerMeterInterface](#) (object)

`value` **(string)** Location of the power meter / sensor, depending on the type of interface ([Path Property](#))

- For **naGPIB**, address of the power meter. Choose any integer between 0 and 30.
- For **naUSB**, the ID string of the power sensor. Use [USBPowerMeterCatalog Property](#) to see a list of ID strings of connected power sensors.
- For **naLAN**, the hostname or IP address of the power meter.
- For **naANY**, any VISA resource string or a VISA alias.

Return Type String

Default Not applicable

Examples `pwrMeterInterface.Locator = "13" 'GPIB address`

```
pwrMeterInterface.Locator = "Agilent
Technologies,U2000A,MY12345678" 'USB ID string
```

```
pwrMeterInterface.Locator = "mymeter.agilent.com" 'LAN
```

```
pwrMeterInterface.Locator =
"TCPIP0::mymeter.agilent.com::5025::SOCKET" ' Any
```

C++ Syntax `HRESULT put_Locator(BSTR pValue);`

`HRESULT get_Locator(BSTR* pValue);`

Locator IPowerMeterInterface

24-Jan-2012 Updated with 'Any'
29-Sep-2008 Removed Rev from Example
24-Jul-2007 MX New topic

Lock Property

Description Enables or disables the ability to change the power limit values through the user interface.

VB Syntax `gpl.Lock = state`

Variable [\(Type\)](#) - Description

gpl A [GlobalPowerLimit](#) (**object**)

state (**boolean**)

False - Disables the ability to change the power limit values from the user interface.

True - Enables the ability to change the power limit values from the user interface.

Return Type Boolean

Default False

Examples `gpl.Lock = True 'Write`

`UIILock = gpl.Lock 'Read`

C++ Syntax HRESULT get_Lock(BOOL *pVal)
HRESULT put_Lock(BOOL newVal)

Interface IGlobalPowerLimit

Last Modified:

5-Aug-2009 MX New topic

LODeltaFound Property

Description Returns the LO frequency delta from the specified tuning sweep.

VB Syntax *value* = *embedLODiag*.**LODeltaFound** (*n*)

Variable [\(Type\)](#) - **Description**

value **(Double)** Variable to store the returned data.

embedLODiag An [EmbeddedLODiagnostic](#) (**object**)

n (Long) Tuning sweep number. Use [NumberOfSweeps](#) to find the number of sweeps taken.

Default Not Applicable

Examples `data= embedLO.LODeltaFound 3 'read`

C++ Syntax HRESULT LODeltaFound(long sweep, double* deltaLO);

Interface IEmbeddedLODiagnostic

Last Modified:

13-Apr-2007 MX New topic

LODenominator Property

Description	Sets or returns the denominator value of the LO Fractional Multiplier. If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.
VB Syntax	<i>obj</i> .LODenominator (<i>n</i>) = <i>value</i>
Variable	(Type) - Description
<i>obj</i>	A Mixer Interface pointer to the Measurement (object) Or A Converter Object
<i>value</i>	(Long) - LO denominator value
<i>n</i>	(Long) - LO stage number Choose from 1 or 2
Return Type	Long
Default	1
Examples	<pre>Print mixer.LODenominator(1) 'prints the value of the first LODenominator</pre>
C++ Syntax	HRESULT get_LODenominator(long *pVal) HRESULT put_LODenominator(long newVal)
Interface	IMixer IConverter

Last Modified:

2-Feb-2009 Added Converter

4-Mar-2008 Added note.

LOFixedFrequency Property

Description Sets or returns the LO frequency value.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

VB Syntax *obj.LOFixedFrequency (n) = value*

Variable [\(Type\)](#) - Description

obj A [Mixer Interface](#) pointer to the [Measurement](#) (object)

Or

A [Converter Object](#)

value **(double)** - LO Frequency in Hertz.

n **(Long)** - LO stage number

Choose from 1 or 2

Return Type Double

Default 0 Hz

Examples

```
Print mixer.LOFixedFrequency(1) 'prints the value of the first
LO fixed frequency
```

C++ Syntax HRESULT get_LOFixedFrequency(double *pVal)

HRESULT put_LOFixedFrequency(double newVal)

Interface IMixer

IConverter

Last Modified:

2-Feb-2009 Added Converter

4-Mar-2008 Added note.

LOFrequencyDelta Property

Description	Sets and returns LO Frequency Delta. There is usually no need to set this value. Read this value to determine the difference between the LO Frequency that is stated in the Mixer dialog box and the last measured LO Frequency.
VB Syntax	<i>obj</i> . LOFrequencyDelta = <i>value</i>
Variable	(Type) - Description
<i>obj</i>	An EmbeddedLO (object) or A ConverterEmbeddedLO (object)
<i>value</i>	(Double) LO Frequency delta in Hertz.
Return Type	(Double)
Default	Not Applicable
Examples	<pre>embedLO.LOFrequencyDelta = 0 'write value = embedLO.LOFrequencyDelta 'read</pre>
C++ Syntax	HRESULT get_LOFrequencyDelta(double* val); HRESULT put_LOFrequencyDelta(double val);
Interface	IEmbeddedLO

Last Modified:

12-Aug-2009 Added ConvEmbedLO object
18-Apr-2007 MX New topic

LogMagnitudeOffset Property **Superseded**

Description **Note:** This property is replaced by [DoReceiverPowerCal Method](#).

For Receiver Calibration - Sets or returns the value to offset the normalized unratiod power measurement data. The unratiod power measurement is effectively calibrated to the power level specified by the value of LogMagnitudeOffset as soon as the [Normalization property](#) is set to ON after calling the [DataToDivisor](#) method.

To offset the data trace magnitude a specified value, use [MagnitudeOffset Property](#)

VB Syntax *meas*.LogMagnitudeOffset = *value*

Variable [\(Type\)](#) - Description

meas **(object)** - A [Measurement](#) object

value **(double)** - Offset value in dBm.

Return Type Double

Default 0

Examples

```
meas.LogMagnitudeOffset = -10 'Write (-10 dBm)
calpower = meas.LogMagnitudeOffset 'Read
meas.DataToDivisor 'Store meas data as measurement divisor
meas.Normalize = 1 'Measurement is now calibrated to -10 dBm
```

C++ Syntax HRESULT put_LogMagnitudeOffset(double newVal);

HRESULT get_LogMagnitudeOffset(double *pVal);

Interface IMeasurement

LOName Property

Description Sets or returns the name of the PNA internal source or external source to use as the LO in an FCA measurement.

VB Syntax *obj.LOName (n) =value*

Variable [\(Type\)](#) - Description

obj A [Mixer Interface](#) pointer to the [Measurement](#) (object)

Or

A [Converter Object](#)

n **(Long)** - LO stage number

Choose from 1 or 2

value **(string)** - LO Source name. Use [SourcePortNames Property](#) to return a list of valid source ports. An external source must be configured and selected to be valid. [Learn more about external source configuration.](#)

Note: If the port is defined by a string name, such as an external source or one of the Source 2 outputs on the 2-port, 2-source PNA-x model, then you must use [cap.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#).

Return Type String

Default "Not controlled"

Examples `mixer.LOName(1) = "MySource"`

C++ Syntax HRESULT get_LOName(string *pVal)

HRESULT put_LOName(string newVal)

Interface MIXer

IConverter

Last Modified:

13-Jan-2012	Fixed SPNames link
2-Feb-2009	Added Converter
24-Apr-2008	Added string names note
23-Jul-2007	Updated for external source config.

LONumerator Property

Description Sets or returns the numerator value of the LO Fractional Multiplier.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

VB Syntax *obj.LONumerator* (*n*) = *value*

Variable [\(Type\)](#) - Description

obj A [Mixer Interface](#) pointer to the [Measurement](#) (object)

Or

A [Converter Object](#)

value **(Long)** - LO denominator value

n **(Long)** - LO stage number

Choose from 1 or 2

Return Type Long

Default 1

Examples

```
Print mixer.LONumerator(1) 'prints the value of the
first LO Numerator
```

C++ Syntax HRESULT get_LONumerator(long *pVal)

HRESULT put_LONumerator(long newVal)

Interface IMixer

IConverter

Last Modified:

2-Feb-2009 Added Converter

4-Mar-2008 Added note.

LOPower Property

Description	Sets or returns the value of LO Power.
VB Syntax	<i>obj.LOPower</i> (<i>n</i>) = <i>value</i>
Variable	(Type) - Description
<i>obj</i>	A Mixer Interface pointer to the Measurement (object)
	Or
	A Converter Object
<i>n</i>	(Long) - LO stage number
	Choose from 1 or 2
<i>value</i>	(double) - LO Power in dBm.
Return Type	Double
Default	-10dBm
Examples	<pre>Print mixer.LOPower(1) 'prints the value of the LO Power</pre>
C++ Syntax	HRESULT get_LOPower(double *pVal) HRESULT put_LOPower(double newVal)
Interface	IMixer IConverter

Last Modified:

2-Feb-2009 added converter

LORangeMode Property

Description Sets or returns the LO sweep mode.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

Note: There is also a [LORangeMode](#) on the Converter Object.

VB Syntax *obj.LORangeMode* (*n*) = *value*

Variable [\(Type\)](#) - Description

obj A [Mixer Interface](#) pointer to the [Measurement](#) (object)

n **(Long)** - LO stage number.

Choose from 1 or 2

value **(Enum as MixerRangeMode)** - LO sweep mode. Choose from:

0 - **mixSwept**

1 - **mixFixed**

Return Type Enum

Default 0 - mixSwept

Examples `mixer.LORangeMode(1)=mixSwept`

C++ Syntax HRESULT get_LORangeMode(long LO, enum *pVal)

HRESULT put_LORangeMode(long LO, enum newVal)

Interface IMixer3

Last Modified:

4-Mar-2008 Added note.

LORangeMode Property

Description Sets or returns the LO sweep mode.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

Note: There is also a [LORangeMode](#) on the Mixer Interface

VB Syntax *obj.LORangeMode* (*n*) = *value*

Variable [\(Type\)](#) - Description

obj A [Converter Object](#)

n **(Long)** - LO stage number.

Choose from 1 or 2

value **(Enum as NARangeMode)** - LO sweep mode. Choose from:

0 - **naSwept**

1 - **naFixed**

Return Type Enum

Default 0 - **naSwept**

Examples `mixer.LORangeMode(1)=naSwept`

C++ Syntax HRESULT get_LORangeMode(long LO, enum *pVal)

HRESULT put_LORangeMode(long LO, enum newVal)

Interface IConverter

Last Modified:

18-Feb-2011 New topic

Loss Property

Description	Sets and Returns the insertion loss for the calibration standard.
VB Syntax	<i>calstd.loss</i> = <i>value</i>
Variable	(Type) - Description
<i>calstd</i>	A CalStandard (object). Use calKit. GetCalStandard to get a handle to the standard.
<i>value</i>	(single) - Insertion loss in Gohms / sec. (Giga Ohms per second of electrical delay)
Return Type	Single
Default	Not Applicable
Examples	<pre>calstd.loss = 3.5 'Write 3.5 Gohms of loss</pre> <pre>stdLoss = calstd.loss 'Read the value of Loss</pre>
C++ Syntax	HRESULT get_Loss(float *pVal) HRESULT put_Loss(float newVal)
Interface	ICalStandard

Loss (Power Segment) Property

Description	Sets or returns the loss value associated with a PowerLossSegment.
VB Syntax	<i>lossSeg</i> . Loss = <i>value</i>
Variable	(Type) - Description
<i>lossSeg</i>	A PowerLossSegment (object)
	A PowerLossSegmentPMAR (object)
<i>value</i>	(double) – Loss value in dB. Loss is entered as a POSITIVE number.
Return Type	Double
Default	0
Examples	<pre>lossSeg.Loss = 0.5 'Write lossVal = lossSeg.Loss 'Read</pre>
C++ Syntax	HRESULT put_Loss(Double newVal); HRESULT get_Loss(Double *pVal);
Interface	IPowerLossSegment IPowerLossSegmentPMAR

Last Modified:

27-Aug-2009	Added PMAR (9.0)
2-Jun-2008	Clarified negative value for loss

LOStage Property

Description Sets or returns the number of LO stages present in the mixer.

VB Syntax *obj.LOStage = value*

Variable [\(Type\)](#) - Description

obj A [Mixer Interface](#) pointer to the [Measurement](#) (object)

Or

A [Converter Object](#)

value (**Long**) - Number of LO stages. Choose from **1** or **2**

Return Type Long

Default 1

Examples `mixer.LOStage = 1 'sets the LO Stage value to 1`

C++ Syntax HRESULT get_LOStage(long *pVal)

HRESULT put_LOStage(long newVal)

Interface IMixer

IConverter

Last Modified:

2-Feb-2009 Added converter

LOStartFrequency Property

Description Sets or returns the LO start frequency value. This command can only be used with SMC (not VMC) measurements.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

VB Syntax *obj.LOStartFrequency (n) = value*

Variable [\(Type\)](#) - Description

obj A [Mixer Interface](#) pointer to the [Measurement](#) (object)

Or

A [Converter Object](#)

value **(double)** - LO Start Frequency in Hertz.

n **(Long)** - LO stage number

Choose from 1 or 2

Return Type Double

Default Not Applicable

Examples

```
Print mixer.LOStartFrequency(1) 'prints the value of the first
LO start frequency
```

C++ Syntax HRESULT get_LOStartFrequency(long id, double *pVal)

HRESULT put_LOStartFrequency(long id, double newVal)

Interface IMixer3

IConverter

Last Modified:

2-Feb-2009 Added Converter

4-Mar-2008 Added note.

LOStartPower Property

Description Sets or returns the Start value of a LO Power sweep.. Also set [indx.SweepType](#) to naIMDLPowerSweep.

VB Syntax *obj.LOStartPower (n) = value*

Variable [\(Type\)](#) - Description

obj A [Converter Object](#)

n **(Long)** - LO stage number. Choose 1

value **(double)** - LO start power in dBm.

Return Type Double

Default -20 dBm

Examples `convtr.LOStartPower(1) = -5 'Sets the LO Power sweep start value`

`start = convtr.LOStartPower(1) 'Reads the start value`

C++ Syntax HRESULT get_LOStartPower(long LOStage, double *pVal)

HRESULT put_LOStartPower(long LOStage, double newVal)

Interface IConverter

Last Modified:

27-Mar-2009 MX New topic

LOStopFrequency Property

Description Sets or returns the LO stop frequency value. This command can only be used with SMC (not VMC) measurements.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

VB Syntax *obj.LOStopFrequency (n) = value*

Variable [\(Type\)](#) - Description

obj A [Mixer Interface](#) pointer to the [Measurement](#) (object)

Or

A [Converter Object](#)

value **(double)** - LO Stop Frequency in Hertz.

n **(Long)** - LO stage number

Choose from 1 or 2

Return Type Double

Default Not Applicable

Examples

```
Print mixer.LOStopFrequency(1) 'prints the value of the first LO stop frequency
```

C++ Syntax HRESULT get_LOStopFrequency(long id, double *pVal)

HRESULT put_LOStopFrequency(long id,double newVal)

Interface IMixer3

IConverter

Last Modified:

2-Feb-2009 Added Converter

4-Mar-2008 Added note.

LOStopPower Property

Description Sets or returns the Stop value of a LO Power sweep. Also set [imdx.SweepType](#) to naIMDLPowerSweep.

VB Syntax *obj.LOStopPower (n) = value*

Variable [\(Type\)](#) - Description

obj A [Converter Object](#)

n **(Long)** - LO stage number. Choose 1

value **(double)** - LO Stop power in dBm.

Return Type Double

Default -10 dBm

Examples `convtr.LOStopPower(1) = -5 'Sets the LO Power sweep Stop value`

`Stop = convtr.LOStopPower(1) 'Reads the Stop value`

C++ Syntax HRESULT get_LOStopPower(long LOStage, double *pVal)

HRESULT put_LOStopPower(long LOStage, double newVal)

Interface IConverter

Last Modified:

27-Mar-2009 MX New topic

LowAmplitude Property

Description Sets and returns the Low amplitude (voltage) of the pulse generator.

VB Syntax *extPulseGen.LowAmplitude* = *value*

Variable [\(Type\)](#) - [Description](#)

extPulseGen An [ExternalPulseGenerator](#) (**object**)

value (**Double**) Pulse Generator low amplitude voltage.

Return Type Double

Default 0

Examples `extPulseGen.LowAmplitude = 4 'Write`

`lo = extPulseGen.LowAmplitude 'Read`

C++ Syntax HRESULT get_LowAmplitude (double *pValue)

HRESULT put_LowAmplitude (double newVal)

Interface IExternalPulseGenerator

Last Modified:

15-Feb-2012 New topic

LXIDeviceIDState Property

Description Opens and closes the [LAN Status dialog](#) with the LAN Status Indicator showing IDENTIFY.

The PNA supports this capability to satisfy a requirement of the LAN eXtensions for Instrumentation (LXI) standard. Changing the value of this property is the same operation that occurs when clicking the Toggle LXI Identification button on the Welcome web page presented by the PNA web server.

VB Syntax `app.LXIDeviceIDState = bool`

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

bool (boolean) Choose from:

True - Displays the [LAN Status dialog](#) with the Status Indicator showing IDENTIFY.

False -

- If the dialog had been opened by this property, then the LAN Status dialog is closed.
- If the dialog was opened manually, then it will stay open.

Return Type Boolean

Default False

Examples

```
app.LXIDeviceIDState = True 'Write
```

```
value = app.LXIDeviceIDState 'Read
```

C++ Syntax HRESULT get_LXIDeviceIDState(VARIANT_BOOL* pState);

HRESULT put_LXIDeviceIDState(VARIANT_BOOL state);

Interface IApplication14

Last Modified:

2-Jun-2008 MX New topic

MagnitudeOffset Property

Description Offsets the data trace magnitude by the specified value.

To offset the data trace magnitude to a slope value that changes with frequency, use [MagnitudeSlopeOffset Property](#).

To implement a Receiver Cal offset, use [LogMagnitudeOffset property](#).

VB Syntax *meas.MagnitudeOffset* = *value*

Variable [\(Type\)](#) - Description

meas **(object)** - A [Measurement](#) object

value **(double)** - Offset value in dB.

Return Type Double

Default 0

Examples

```
meas.MagnitudeOffset = 4 'Write
```

```
offs = meas.MagnitudeOffset 'Read
```

C++ Syntax HRESULT put_MagnitudeOffset(double newVal);

HRESULT get_MagnitudeOffset(double *pVal);

Interface IMeasurement4

MagnitudeSlopeOffset Property

Description Offsets the data trace magnitude to a value that changes linearly with frequency. The offset slope begins at 0 Hz.

To offset the entire data trace magnitude by a specified value, use [MagnitudeOffset Property](#).

VB Syntax *meas.MagnitudeSlopeOffset = value*

Variable [\(Type\)](#) - Description

meas **(object)** - A [Measurement](#) object

value **(double)** - Offset slope value in dB / 1GHz.

Return Type Double

Default 0

Examples

```
meas.MagnitudeSlopeOffset = 4 'Writes a slope offset of 4dB/1GHz.  
offs = meas.MagnitudeSlopeOffset 'Read
```

C++ Syntax HRESULT put_MagnitudeSlopeOffset(double newVal);

HRESULT get_MagnitudeSlopeOffset(double *pVal);

Interface IMeasurement4

MainToneIFBandwidth Property

Description Sets and returns the IF Bandwidth for measurement of the Main tones.

VB Syntax `imd.MainToneIFBandwidth = value`

Variable [\(Type\)](#) - Description

`imd` A [SweptIMD](#) Object

`value` (Double) IF Bandwidth in Hz. Choose from: 1 | 2 | 3 | 5 | 7 | 10 | 15 | 20 | 30 | 50 | 70 | 100 | 150 | 200 | 300 | 500 | 700 | 1k | 1.5k | 2k | 3k | 5k | 7k | 10k | 15k | 20k | 30k | 50k | 70k | 100k | 150k | 200k | 280k | 360k | 600k

[Learn more about setting IFBW for IMD.](#)

If an invalid number is specified, the analyzer will round up to the closest valid number.

Return Type Double

Default 1 kHz

Examples

```
imd.MainToneIFBandwidth = 2e3 'Write
```

```
value = imd.MainToneIFBandwidth 'Read
```

C++ Syntax HRESULT get_MainToneIFBandwidth(double *pVal)
HRESULT put_MainToneIFBandwidth(double newVal)

Interface ISweptIMD

Last Modified:

19-Aug-2008 MX New topic

MarkerAnnotation Property

Description Returns the Y-axis marker value from the specified tuning sweep. Use [IsMarkerOn](#) to confirm if a marker was used for the tuning sweep.

VB Syntax *value* = *embedLO*.MarkerAnnotation (*n*)

Variable [\(Type\)](#) - Description

value **(String)** Variable to store the returned data.

embedLO An [EmbeddedLODiagnostic](#) (**object**)

n (Long) Tuning sweep number. Use [NumberOfSweeps](#) to find the number of sweeps taken.

Default Not Applicable

Examples `data= embedLO.MarkerAnnotation 3 'read`

C++ Syntax HRESULT MarkerAnnotation(long sweep, BSTR* annotation);

Interface IEmbeddedLODiagnostic

Last Modified:

13-Apr-2007 MX New topic

MarkerFormat Property

Description Sets the format of all the markers in the measurement. To override this setting for an individual marker, use [mark.Format](#)

VB Syntax `meas.MarkerFormat = value`

Variable [\(Type\)](#) - Description

meas A Measurement (**object**)

value (**enum NAMarkerFormat**) - Choose from:

0 - naMarkerFormat_LinMag

1 - naMarkerFormat_LogMag

2 - naMarkerFormat_Phase

3 - naMarkerFormat_Delay

4 - naMarkerFormat_Real

5 - naMarkerFormat_Imaginary

6 - naMarkerFormat_SWR

7 - naMarkerFormat_LinMagPhase

8 - naMarkerFormat_LogMagPhase

9 - naMarkerFormat_RealImaginary

10 - naMarkerFormat_ComplexImpedance

11 - naMarkerFormat_ComplexAdmittance

12 - naMarkerFormat_Kelvin

13 - naMarkerFormat_Fahrenheit

14 - naMarkerFormat_Centigrade

Return Type Not Applicable

Default 1 - naMarkerFormat_LogMag

Examples `meas.MarkerFormat = naMarkerFormat_SWR 'Write`

[C++ Syntax](#) HRESULT put_MarkerFormat(tagNAMarkerFormat NewFormat)

[Interface](#) IMeasurement

Last Modified:

17-Nov-2010	Removed Read capability
1-Oct-2007	Added temperature formats

InterpolateMarkers Method

Description Turns **All** Marker Interpolation ON and OFF for the measurement. Marker interpolation enables X-axis resolution between the discrete data values. The analyzer will calculate the x and y-axis data values between discrete data points. To override this property for individual markers, use the [Interpolated](#) property.

VB Syntax *meas*.Interpolate = *value*

Variable [\(Type\)](#) - Description

meas A Measurement (**object**)

value (**boolean**)

False - Turns interpolation OFF for all markers in the measurement

True - Turns interpolation ON for all markers in the measurement

Return Type Boolean

Default True (ON)

Examples `meas.Interpolate = 1`

C++ Syntax HRESULT InterpolateMarkers(VARIANT_BOOL bNewVal)

Interface IMeasurement

Number Property

Description	Returns the number of the marker.
VB Syntax	<i>marknum</i> = <i>mark</i> . Number
Variable	(Type) - Description
<i>marknum</i>	(long) - Variable to store marker number
<i>mark</i>	A Marker (object)
Return Type	Long Integer
Default	Not applicable
Examples	<pre>marknum = mark.Number 'Read</pre>
C++ Syntax	HRESULT get_Number(long *pVal)
Interface	IMarker

MarkerPosition Property

Description Returns the X-axis marker position from the specified tuning sweep.

VB Syntax *value* = *embedLO*.**MarkerPosition** (*n*)

Variable [\(Type\)](#) - Description

value **(Double)** Variable to store the returned data.

embedLO *Diag* An [EmbeddedLODiagnostic](#) (**object**)

n (Long) Tuning sweep number. Use [NumberOfSweeps](#) to find the number of sweeps taken.

Default Not Applicable

Examples `data= embedLO.MarkerPosition 3 'read`

C++ Syntax HRESULT MarkerPosition(long sweep, double *position);

Interface IEmbeddedLODiagnostic

Last Modified:

13-Apr-2007 MX New topic

MarkerReadout Property

Description	Enables or disables the readout of markers in the window. To show the marker on the screen use ShowMarkerReadout Method.
VB Syntax	<i>win</i> . MarkerReadout = <i>state</i>
Variable	(Type) - Description
<i>win</i>	A NAWindow (object)
<i>state</i>	(boolean) True - enables marker readout False - disables marker readout
Return Type	Boolean
Default	True
Examples	<pre>win.MarkerReadout = True 'Write</pre> <pre>State = app.ActiveNAWindow.MarkerReadout 'Read</pre>
C++ Syntax	HRESULT get_MarkerReadout(VARIANT_BOOL *pVal) HRESULT put_MarkerReadout(VARIANT_BOOL newVal)
Interface	INAWindow

MarkerReadoutResponsePlaces Property

Description	For the Y-axis (response), sets the number digits to display after the decimal point in marker readouts.
VB Syntax	<i>win.MarkerReadoutResponsePlaces = value</i>
Variable	(Type) - Description
<i>win</i>	A NAWindow (object)
<i>value</i>	(Long Integer) Number of digits to display. Choose a value between 1 and 4.
Return Type	Long Integer
Default	2
Examples	<pre>win.MarkerReadoutResponsePlaces = 3 'Write</pre> <pre>value = app.ActiveNAWindow.MarkerReadoutResponsePlaces 'Read</pre>
C++ Syntax	HRESULT get_MarkerReadoutResponsePlaces(long *pVal) HRESULT put_MarkerReadoutResponsePlaces(long newVal)
Interface	INAWindow3

Last Modified:

5-Aug-2010 MX New topic

MarkerReadoutSize Property

Description	Specifies the size of font used when displaying Marker Readout in the selected window.
VB Syntax	<i>win</i> . MarkerReadoutSize = <i>value</i>
Variable	(Type) - Description
<i>win</i>	A NAWindow (object)
<i>value</i>	(enum NAFontSize)
	0 - naDefault - marker readout appears in default font size
	1 - naLarge - marker readout appears in large font size
Return Type	Long Integer
Default	naDefault
Examples	<pre>win.MarkerReadoutSize = naDefault 'write default size for marker readout</pre>
	<pre>Dim Size As NAFontSize Size = app.ActiveNAWindow.MarkerReadoutSize 'Read</pre>
C++ Syntax	HRESULT get_MarkerReadoutSize(tagNAFontSize *pVal) HRESULT put_MarkerReadoutSize(tagNAFontSize newVal)
Interface	INAWindow

MarkerReadoutsPerTrace Property

Description Sets the number of marker readouts to display per trace. Display up to 20 marker readouts per window.

VB Syntax `win.MarkerReadoutsPerTrace = value`

Variable [\(Type\)](#) - Description

win A [NAWindow](#) (object)

value **(Long Integer)** Number of marker readouts to display. Choose a value between 1 and 10.

Return Type Long Integer

Default 5

Examples `win.MarkerReadoutsPerTrace = 3 'Write`

`value = app.ActiveNAWindow.MarkerReadoutsPerTrace 'Read`

C++ Syntax HRESULT get_MarkerReadoutsPerTrace(long *pVal)

HRESULT put_MarkerReadoutsPerTrace(long newVal)

Interface INAWindow3

Last Modified:

5-Aug-2010 MX New topic

MarkerReadoutStimulusPlaces Property

Description	For the X-axis (stimulus), sets the number digits to display after the decimal point in marker readouts.
VB Syntax	<i>win.MarkerReadoutStimulusPlaces = value</i>
Variable	(Type) - Description
<i>win</i>	A NAWindow (object)
<i>value</i>	(Long Integer) Number of digits to display. Choose a value between 2 and 6.
Return Type	Long Integer
Default	3
Examples	<pre>win.MarkerReadoutStimulusPlaces = 2 'Write</pre> <pre>value = app.ActiveNAWindow.MarkerReadoutStimulusPlaces 'Read</pre>
C++ Syntax	HRESULT get_MarkerReadoutStimulusPlaces(long *pVal) HRESULT put_MarkerReadoutStimulusPlaces(long newVal)
Interface	INAWindow3

Last Modified:

5-Aug-2010 MX New topic

MarkerReadoutXPosition Property

Description Sets the X-axis position of marker readouts. Readouts are right-justified at the specified position.

VB Syntax `win.MarkerReadoutXPosition = value`

Variable [\(Type\)](#) - Description

win A [NAWindow](#) (object)

value **(Double)** X-axis position. Choose a value between 1 (far left) and 10 (far right).

Return Type Double

Default 10

Examples `win.MarkerReadoutXPosition = 3 'Write`

`value = app.ActiveNAWindow.MarkerReadoutXPosition 'Read`

C++ Syntax HRESULT get_MarkerReadoutXPosition(double *pVal)

HRESULT put_MarkerReadoutXPosition(double newVal)

Interface INAWindow3

Last Modified:

5-Aug-2010 MX New topic

MarkerReadoutYPosition Property

Description	Sets the Y-axis position of marker readouts. Readouts are right-justified at the specified position.
VB Syntax	<i>win</i> . MarkerReadoutYPosition = <i>value</i>
Variable	(Type) - Description
<i>win</i>	A NAWindow (object)
<i>value</i>	(Double) Y-axis position. Choose a value between 1 (bottom) and 10 (top).
Return Type	Double
Default	10
Examples	<pre>win.MarkerReadoutYPosition = 3 'Write</pre> <pre>value = app.ActiveNAWindow.MarkerReadoutYPosition 'Read</pre>
C++ Syntax	HRESULT get_MarkerReadoutYPosition(double *pVal) HRESULT put_MarkerReadoutYPosition(double newVal)
Interface	INAWindow3

Last Modified:

5-Aug-2010 MX New topic

Markers Property

Description Set and return the color of data trace markers for nth trace in a window.

VB Syntax `trace(n).Markers = value`

Variable [\(Type\) - Description](#)

`trace(n)` One of the 8 [ComTraceColors](#) objects

`value` **(Long Integer)** - RGB color of the Markers pen.

Convert the three RGB colors to an integer as follows:

```
RGB = R+(G*2^8)+(B*2^16)
```

To find the three RGB values from the [Display Colors dialog](#), click **Change Color**, then **Define Custom Color**.

Return Type Long

Default Varies for each trace.

Examples

```
R = 10
G = 10
B = 10
RGB = 10+(10*2^8)+(10*2^16)
trace1.Markers = RGB 'Write
color = trace1.Markers 'Read
```

C++ Syntax HRESULT get_Markers(long* pVal);

HRESULT put_Markers(long newVal);

Interface IComTraceColors

Last Modified:

7-Aug-2009 MX New topic

MarkerState Property

Description Sets or returns the ON / OFF state of the specified marker.

VB Syntax `meas.MarkerState (n) = state`

Variable [\(Type\)](#) - Description

meas A Measurement (**object**)

n (Long Integer) Marker number to turn on or off.

state (boolean) -
True - turns the specified marker ON
False - turns the specified marker OFF

Return Type Boolean

Default False

Examples `meas.MarkerState(1) = True`

`reference = meas.MarkerState(2)`

C++ Syntax HRESULT get_MarkerState(long markerNum, VARIANT_BOOL bState)
HRESULT put_MarkerState(long markerNum, VARIANT_BOOL* bState)

Interface IMeasurement3

MarkerSymbol Property

Description Sets the symbol to display for marker position.

VB Syntax `win.MarkerSymbol = value`

Variable [\(Type\)](#) - Description

win A [NAWindow](#) (object)

value (Enum as naMarkerSymbol) Choose from:

0 - naMarkerSymbolTriangle

1 - naMarkerSymbolFlag

2 - naMarkerSymbolLine

[See pictures of symbols](#)

Return Type Enum

Default 0 - naMarkerSymbolTriangle

Examples

```
win.MarkerSymbol = naMarkerSymbolLine  
'Write
```

```
value = app.ActiveNAWindow.MarkerSymbol  
'Read
```

C++ Syntax HRESULT get_MarkerSymbol(double *pVal)

HRESULT put_MarkerSymbol(double newVal)

Interface INAWindow3

Last Modified:

5-Aug-2010 MX New topic

MasterFrequency Property

Description Sets the pulse repetition frequency (PRF) for ALL internal pulse generators.

This setting is equal to 1/period which is set with [MasterPeriod Property](#)

Note: On the [Pulse Setup dialog](#), this command is a "Basic" setting, intended to be used with the 'Auto' selections set to ON.

VB Syntax *pulse*.**MasterFrequency** = *value*

Variable [\(Type\)](#) - Description

pulse A [PulseMeasurementControl](#) (**object**)

value (**Double**) PRF in Hz.

Return Type Double

Default 1 kHz

Examples `pulse.MasterFrequency = 1e4 'Write`
`value = pulse.MasterFrequency 'Read`

C++ Syntax HRESULT get_MasterFrequency(double* value);
HRESULT put_MasterFrequency(double value);

Interface IPulseMeasurementControl2

Last modified:

11-May-2011 New topic

MasterMode Property

Description Sets and returns the Master (On/Off) setting of the external pulse generator. this setting allows the external pulse generator to set the master clock frequency for the other pulse generators.

VB Syntax `extPulseGen.MasterMode = value`

Variable [\(Type\)](#) - **Description**

extPulseGen An [ExternalPulseGenerator](#) (**object**)

value (**Boolean**) Master setting. Choose from:

True - Use the external pulse generator becomes the master clock frequency.

False - Use the internal pulse generator as the master clock frequency.

Return Type Boolean

Default False

Examples

```
extPulseGen.MasterMode = True 'Write
```

```
master = extPulseGen.MasterMode 'Read
```

C++ Syntax HRESULT get_MasterMode (VARIANT BOOL *pValue)

HRESULT put_MasterMode (VARIANT BOOL newVal)

Interface IExternalPulseGenerator2

Last Modified:

15-Feb-2012 New topic

MasterPeriod Property

Description Sets the period for ALL internal pulse generators.

This setting is equal to 1/PRF which is set with [MasterFrequency Property](#).

Note: On the [Pulse Setup dialog](#), this command is a 'Basic setting, which is intended to be used with the 'Auto' selections set to ON.

VB Syntax *pulse*.**MasterPeriod** = *value*

Variable [\(Type\)](#) - Description

pulse A [PulseMeasurementControl Object](#) (**object**)

value (**Double**) Period in seconds.

Return Type Double

Default 1 msec

Examples
`pulse.MasterPeriod = 1e-4 'Write`
`value = pulse.MasterPeriod 'Read`

C++ Syntax HRESULT get_MasterPeriod(double* MasterPeriod);
HRESULT put_MasterPeriod(double MasterPeriod);

Interface IPulseMeasurementControl2

Last modified:

9-May-2011 New topic

MasterWidth Property

Description Sets the pulse width for ALL internal pulse generators.

Note: On the [Pulse Setup dialog](#), this command is a 'Basic setting, which is intended to be used with the 'Auto' selections set to ON.

VB Syntax `pulse.MasterWidth = value`

Variable [\(Type\)](#) - Description

pulse A [PulseMeasurementControl Object](#) (**object**)

value (**Double**) Pulse width in seconds.

Return Type Double

Default 100 microseconds

Examples `pulse.MasterWidth = 1e-3 'Write`

`value = pulse.MasterWidth 'Read`

C++ Syntax HRESULT get_MasterWidth(double* MasterWidth);

HRESULT put_MasterWidth(double MasterWidth);

Interface IPulseMeasurementControl2

Last modified:

9-May-2011 New topic

Type (Marker) Property

Description Sets and reads the marker type.

VB Syntax *mark.Type = value*

Variable [\(Type\)](#) - Description

chan A [Marker](#) (object)

value (enum `NAMarkerType`) - Marker Type. Choose from:

0 - `naMarkerType_Normal` - the X-axis value for a normal marker will always be determined by the measurement data of the marker.

1 - `naMarkerType_Fixed` - retains and keeps its x-axis value at the time the marker type is set.

Return Type Long Integer

Default `naMarkerType_Normal`

Examples `mark.Type = naMarkerType_Normal 'Write`

`MrkType = mark.Type 'Read`

C++ Syntax `HRESULT get_Type(tagNAMarkerType *pVal)`
`HRESULT put_Type(tagNAMarkerType newVal)`

Interface `IMarker`

Stimulus Property

Description Sets and reads the X-Axis value of the marker. If the marker is a delta marker, the value will be relative to the reference marker.

VB Syntax *mark.Stimulus = value*

Variable [\(Type\)](#) - Description

mark A [Marker](#) (object)

value (**double**) - X-Axis value. Choose any number within the full span of the channel or User Range (if set).

Return Type Double

Default First activated Marker turns ON in the middle of the X-axis range. Subsequent markers turn ON at the position of the most recently active marker.

Examples `mark.Stimulus = 3e9 'Write`

`XVal = mark.Stimulus 'Read`

C++ Syntax HRESULT get_Stimulus(double *pVal)
HRESULT put_Stimulus(double newVal)

Interface IMarker

Value Property

Description Reads the Y-axis value of the marker. If the marker is a delta marker, the value will be relative to the reference marker.

You cannot set the Y-axis value of a marker. The marker remains at the position at the time you set [marker.Type](#).

Note: To accurately read the marker Y-axis value with [trace smoothing](#) applied, the requested format must match the [displayed format](#). Otherwise, the returned value is un-smoothed data. For example, to read the smoothed marker value when measuring group delay, both the display format and the marker format must be set to (Group) Delay.

VB Syntax *YValue* = *mark.Value* (*format*)

Variable [\(Type\)](#) - **Description**

YValue A variable to store the Y-axis value

mark A [Marker](#) (**object**)

format (**enum NAMarkerFormat**) - The format in which to return the marker's Y-axis value. The number in parenthesis following the format is the number of values that are returned in a variant array. Choose from:

0 - naMarkerFormat_**LinMag** (1)

1 - naMarkerFormat_**LogMag** (1)

2 - naMarkerFormat_**Phase** (1)

3 - naMarkerFormat_**Delay** (1)

4 - naMarkerFormat_**Real** (1)

5 - naMarkerFormat_**Imaginary** (1)

6 - naMarkerFormat_**SWR** (1)

7 - naMarkerFormat_**LinMagPhase** (2)

8 - naMarkerFormat_**LogMagPhase** (2)

9 - naMarkerFormat_**RealImaginary** (2)

10 - naMarkerFormat_**ComplexImpedance** (3)

11 - naMarkerFormat_ **ComplexAdmittance** (3)

12 - naMarkerFormat_ **Kelvin** (1)

13 - naMarkerFormat_ **Fahrenheit** (1)

14 - naMarkerFormat_ **Celsius** (1)

Return Type Variant - The (parens) in the previous list of formats indicates the number of values that are returned in a variant array

Default Not applicable

Examples

```
YVal = mark.Value(0) 'Read  
'or  
YVal = mark.Value(naMarkerFormat_LinMag)
```

C++ Syntax HRESULT get_Value(tagNAMarkerFormat format, VARIANT *pVal)

Interface IMarker

Last modified:

13-Apr-2011 Fixed example
1-Oct-2007 Added temperature formats
4-Dec-2006 Added smoothing note

Read/Write

MatchCorrectPower Property

Description Turns match-correction ON or OFF. Use this command AFTER performing a Guided Power Cal. [Learn more.](#)

VB Syntax *corrMethods.MatchCorrectPower = value*

Variable [\(Type\) - Description](#)

corrMethods [CorrectionMethods](#) (object)

value **(Boolean)**

True Turns match-correction ON

False Turns match-correction OFF

Return Type **Boolean**

Default True

Example `corrMethods.MatchCorrectPower = True`

C++ Syntax HRESULT get_MatchCorrectPower(VARIANT_BOOL* val);

HRESULT put_MatchCorrectPower(VARIANT_BOOL newVal);

Interface ICorrectionMethods

Last Modified:

10-Sep-2010 MX New topic

MaximumFrequency Property

Description Sets and Returns the maximum frequency for the calibration standard.

VB Syntax *calstd.MaximumFrequency = value*

Variable [\(Type\)](#) - Description

calstd A CalStandard (**object**). Use calKit.[GetCalStandard](#) to get a handle to the standard.

value (**double**) - Maximum frequency in Hertz.

Return Type Double

Default Not Applicable

Examples `calstd.MaximumFrequency = 9e9 'Write`

`maxFrequency = calstd.MaximumFrequency 'Read`

C++ Syntax HRESULT get_MaximumFrequency(double *pVal)
HRESULT put_MaximumFrequency(double newVal)

Interface ICalStandard

MaximumFrequency Property

Description Returns the maximum frequency of the remote PNA.

VB Syntax *value* = *cap*.MaximumFrequency

Variable [\(Type\)](#) - Description

value (Double) - Variable to store the returned maximum frequency of the PNA.

cap A [Capabilities](#) (object)

Return Type Double

Default Not Applicable

Examples `value = cap.MaximumFrequency 'Read`

C++ Syntax HRESULT get_MaximumFrequency(&frequencyMax);

Interface ICapabilities

MaximumFrequency Property

Description Set and returns the maximum usable frequency specified for the power sensor.

VB Syntax *pwrSensor*.**MaximumFrequency** = *value*

Variable [\(Type\)](#) - Description

pwrSensor A [PowerSensor](#) (Object) or

a [PowerSensorAsReceiver](#) (Object)

value (**double**) -Frequency in Hertz.

Return Type Double

Default Device dependent

Examples `pwrSensor.MaximumFrequency = 6e9 'Write`

`MaxFreq = pwrSensor.MaximumFrequency 'Read`

C++ Syntax HRESULT put_MaximumFrequency(double newVal);

HRESULT get_MaximumFrequency(double *pVal);

Interface IPowerSensor

IPowerSensorPMAR

MaximumIFFrequency Property

Description	Returns the Maximum allowed value for the IFFrequency Property
VB Syntax	<i>value</i> = <i>IfConfig</i> . MaximumIFFrequency
Variable	(Type) - Description
<i>value</i>	(double) Variable to store the returned maximum allowed frequency that can be applied to the IFFrequency Property .
<i>IfConfig</i>	IFConfiguration (object)
Return Type	Double
Default	Not Applicable
Examples	<pre>val = App.ActiveChannel.IFConfiguration.MaximumIFFrequency 'Read</pre>
C++ Syntax	HRESULT get_ MaximumIFFrequency (double * pMaxFreq);
Interface	IIFConfiguration3

Last Modified:

18-Jun-2007 MX New topic

MaximumIterationsPerPoint Property

Description This command, along with [IterationsTolerance Property](#) deal with ADJUSTMENTS made to the source power.

Sets the maximum number of readings to take at each data point for iterating the source power. Power readings will continue to be made, and source power adjusted, until a reading is within the [IterationsTolerance](#) value or this max number of readings has been met. The last value to be read is the valid reading for that data point.

The following two commands allow for settling of power READINGS.

[ReadingsPerPoint Property](#)

[ReadingsTolerance Property](#)

VB Syntax `pwrCal.MaximumIterationsPerPoint = value`

Variable [\(Type\)](#) - Description

`pwrCal` **(object)** - A [SourcePowerCalibrator](#) (object)

`value` **(Long)** – Maximum number of readings. Choose any number between 1 and 100.

Return Type Long Integer

Default 5

Examples

```
Set powerCalibrator = pna.SourcePowerCalibrator
powerCalibrator.MaximumIterationsPerPoint = 5 'Write
MaxReads = powerCalibrator.MaximumIterationsPerPoint 'Read
```

C++ Syntax HRESULT get_MaximumIterationsPerPoint(long *pVal);

HRESULT put_MaximumIterationsPerPoint(long newVal);

Interface ISourcePowerCalibrator3

Last Modified:

17-Apr-2007 Clarified verbage

MaximumNumberOfChannels Property

Description	Returns the maximum possible number of channels that can be used in the PNA.
VB Syntax	<i>value</i> = <i>cap</i> .MaximumNumberOfChannels
Variable	(Type) - Description
<i>value</i>	(Long) - Variable to store the returned maximum value for number of channels.
<i>cap</i>	A Capabilities (object)
Return Type	Long
Default	Not Applicable
Examples	<pre>value = cap.MaximumNumberOfChannels 'Read</pre>
C++ Syntax	HRESULT get_MaximumNumberOfChannels(long * maximumNumberOfChans);
Interface	ICapabilities2

MaximumNumberOfTracesPerWindow Property

Description	Returns the maximum possible number of traces that can reside in any window.
VB Syntax	<i>value</i> = <i>cap</i> . MaximumNumberOfTracesPerWindow
Variable	(Type) - Description
<i>value</i>	(Long) - Variable to store the returned maximum value for number of traces.
<i>cap</i>	A Capabilities (object)
Return Type	Long
Default	Not Applicable
Examples	<pre>value = cap.MaximumNumberOfTracesPerWindow 'Read</pre>
C++ Syntax	HRESULT get_MaximumNumberOfTracesPerWindow(long * maximumNumberOfTraces);
Interface	ICapabilities2

MaximumNumberOfWindows Property

Description	Returns the maximum possible number of windows that can be present on the PNA screen.
VB Syntax	<i>value</i> = <i>cap</i> .MaximumNumberOfWindows
Variable	(Type) - Description
<i>value</i>	(Long) - Variable to store the returned maximum value for number of windows.
<i>cap</i>	A Capabilities (object)
Return Type	Long
Default	Not Applicable
Examples	<pre>value = cap.MaximumNumberOfWindows 'Read</pre>
C++ Syntax	HRESULT get_MaximumNumberOfWindows(long * maximumNumberOfWindows);
Interface	ICapabilities2

MaximumNumberOfPoints Property

Description	Returns the maximum possible number of data points.
VB Syntax	<i>value</i> = <i>obj</i> . MaximumNumberOfPoints
Variable	(Type) - Description
<i>value</i>	(Long) - Variable to store the returned maximum value for number of points.
<i>cap</i>	A Capabilities (object)
	or
	A GainCompression (object)
Return Type	Double
Default	Not Applicable
Examples	<pre>value = cap.MaximumNumberOfPoints 'Read</pre>
C++ Syntax	HRESULT get_MaximumNumberOfPoints(double * maximumNumberOfPoints);
Interface	ICapabilities IGainCompression

Last Modified:

24-Oct-2013 Changed Return Type
3-Dec-2007 MX New topic

MaximumPowerCompression

MaximumReceiverStepAttenuator Property

Description	Returns the maximum amount of receiver attenuation.
VB Syntax	<i>value</i> = <i>cap</i> . MaximumReceiverStepAttenuator (<i>n</i>)
Variable	(Type) - Description
<i>value</i>	(Double) - Variable to store the returned value of maximum receiver attenuation.
<i>cap</i>	A Capabilities (object)
<i>n</i>	(Long) - port number to query for step attenuators
Return Type	Double
Default	Not Applicable
Examples	<code>value = cap.MaximumReceiverStepAttenuator 'Read</code>
C++ Syntax	HRESULT get_MaximumReceiverStepAttenuator(long portNumber, double * attenuation);
Interface	ICapabilities

MaximumSourceALCPower Property

Description	Returns a value indicating the maximum amount of source ALC power.
VB Syntax	<i>value</i> = <i>cap</i> .MaximumSourceALCPower (<i>n</i>)
Variable	(Type) - Description
<i>value</i>	(Double) - Variable to store the returned value for the maximum amount of source ALC power.
<i>cap</i>	A Capabilities (object)
<i>n</i>	(Long) - source number to query for maximum ALC power
Return Type	Double
Default	Not Applicable
Examples	<pre>value = cap.MaximumSourceALCPower 'Read</pre>
C++ Syntax	HRESULT get_MaximumSourceALCPower(long sourceNum, double * power);
Interface	ICapabilities

MaximumSourceStepAttenuator Property

Description	Returns a value for the maximum amount of source attenuation.
VB Syntax	<i>value</i> = <i>cap</i> .MaximumSourceStepAttenuator (<i>n</i>)
Variable	(Type) - Description
<i>value</i>	(Double) - Variable to store the returned value for the maximum amount of source attenuation.
<i>cap</i>	A Capabilities (object)
<i>n</i>	(Long) - port number to query for the maximum amount of source attenuation
Return Type	Double
Default	Not Applicable
Examples	<pre>value = cap.MaximumSourceStepAttenuator 2 'Read</pre>
C++ Syntax	HRESULT get_MaximumSourceStepAttenuator(long portNumber, double * attenuation);
Interface	ICapabilities

Last Modified:

19-Sep-2007 Added port arg to ex.

MaxPreciseTuningIterations Property

Description Sets and returns the maximum number of tuning iterations to achieve the precise tolerance.

VB Syntax *obj*.MaxPreciseTuningIterations = *value*

Variable [\(Type\)](#) - Description

obj An [EmbeddedLO](#) (**object**) or

A [ConverterEmbeddedLO](#) (**object**)

value (**Long**) Maximum number of tuning iterations.

Return Type (**Long**)

Default 5

Examples

```
embedLO.MaxPreciseTuningIterations = 3 'write
```

```
value = embedLO.MaxPreciseTuningIterations 'read
```

C++ Syntax HRESULT get_MaxPreciseTuningIterations long* iter);

HRESULT put_MaxPreciseTuningIterations long iter);

Interface IEmbeddedLO

Last Modified:

12-Aug-2009 Added ConvEmbedLO object

18-Apr-2007 MX New topic

MaxProduct Property

Description Sets and returns the maximum intermod product frequencies to be calibrated.

VB Syntax `imd.MaxProduct = value`

Variable [\(Type\)](#) - Description

imd A [SweptIMDCal](#) (object)

value Maximum IM products to calibrate. Choose from:

2 - second order products

3 - third order products

5 - fifth order products

7 - seventh order products

9 - ninth order products

Return Type Long Integer

Default 3

Examples

```
imd.MaxProduct = 7 'Write
```

```
mprod = imd.MaxProduct 'Read
```

C++ Syntax HRESULT get_MaxProduct(long * Val)

HRESULT put_MaxProduct(long newVal)

Interface ISweptIMD

Last Modified:

9-Sep-2008 MX New topic

Mean Property

Description	Returns the mean value of the measurement . To retrieve all 3 statistics value at the same time, use meas.GetTraceStatistics
VB Syntax	<code>average = meas.Mean</code>
Variable	(Type) - Description
<code>average</code>	(single) - Variable to store mean value
<code>meas</code>	A Measurement (object)
Return Type	Single
Default	Not applicable
Examples	<pre>Dim average as Single average = meas.Mean 'Read</pre>
C++ Syntax	HRESULT get_Mean(float* mean)
Interface	IMeasurement

Measurement Property

Description	Returns the measurement handle of the trace object. Learn the difference between a Trace and a Measurement .
VB Syntax	<i>myMeas</i> = <i>trace</i> . Measurement
Variable	(Type) - Description
<i>myMeas</i>	(Object) Variable to store the returned Measurement object.
<i>trace</i>	A Trace (object)
Return Type	Object
Default	Not Applicable
Examples	<code>myMeas = myTrace.Measurement</code>
C++ Syntax	HRESULT get_Measurement(IMeasurement** ppMeas);
Interface	ITrace2

Last modified:

2-May-2011 New topic

MeasurementClassProperties Property

Description Provides access to the [MeasurementClassProperties](#) Object.

VB Syntax *handle* = *cap*.**MeasurementClassProperties** (*measClass*)

Variable [\(Type\)](#) - Description

handle (Variant) - Variable to store the returned object.

cap A [Capabilities](#) (**object**)

measClass (String) Name of the measurement class to be accessed.

Use [AvailableMeasurementClasses Property](#) to return a list of measurement classes installed on the PNA.

Return Type Object

Default Not Applicable

Examples

```
'Access the MeasurmentClassProperties Object
Set app = CreateObject("AgilentPNA835x.Application")
Set cap = app.Capabilities
Set measProps = cap.MeasurmentClassProperties("SweptIMD")
```

C++ Syntax HRESULT put_MeasurementClassProperties(IMeasurementClassProperties *handle);

Interface ICapabilities8

Last Modified:

24-May-2011 MX New topic

MeasurementClass Property

Description	Returns the measurement class name from the channel. Use CreateCustomMeasurementEx to create a measurement from a class other than standard S-Parameters.
VB Syntax	<i>class</i> = <i>chan</i> . MeasurementClass
Variable	(Type) - Description
	<i>class</i> (string) - Variable to store the returned measurement class name.
	<i>chan</i> Channel (object)
Return Type	String
Default	Not Applicable
Examples	<pre>class = chan.MeasurementClass 'Read For a standard S-Parameter channel, returns... "Standard"</pre>
C++ Syntax	HRESULT get_MeasurementClass();
Interface	IChannel15

Last Modified:

6-Oct-2008 Corrected IChannel
12-May-2008 MX New topic

Medium Property

Description Sets and Returns the media type of the calibration standard.

VB Syntax `calstd.Medium = value`

Variable [\(Type\)](#) - Description

calstd A CalStandard (**object**). Use calKit.[GetCalStandard](#) to get a handle to the standard.

value (**enum** NACalStandardMedium) - Medium of the transmission line of the standard.

Choose from:

0 - naCoax - Coaxial Cable

1 - naWaveGuide

Return Type Long Integer

Default Not Applicable

Examples

```
calstd.Medium = naCoax 'Write
```

```
stdMedium = calstd.Medium 'Read
```

C++ Syntax

```
HRESULT get_Medium(tagNACalStandardMedium *pVal)
```

```
HRESULT put_Medium(tagNACalStandardMedium newVal)
```

Interface ICalStandard

Memory Property

Description Set and return the memory trace color for nth trace in a window.

VB Syntax `trace(n).Memory = value`

Variable [\(Type\) - Description](#)

`trace(n)` One of the 8 [ComTraceColors](#) objects

`value` **(Long Integer)** - RGB color of the Memory pen.

Convert the three RGB colors to an integer as follows:

```
RGB = R+(G*2^8)+(B*2^16)
```

To find the three RGB values from the [Display Colors dialog](#), click **Change Color**, then **Define Custom Color**.

Return Type Long

Default Varies for each trace.

Examples

```
R = 10
G = 10
B = 10
RGB = 10+(10*2^8)+(10*2^16)
trace1.Memory = RGB 'Write
color = trace1.Memory 'Read
```

C++ Syntax HRESULT get_Memory(long* pVal);

HRESULT put_Memory(long newVal);

Interface IComTraceColors

Last Modified:

7-Aug-2009 MX New topic

MemoryMarkers Property

Description Set and return the color of memory trace markers for nth trace in a window.

VB Syntax `trace(n).MemoryMarkers = value`

Variable [\(Type\)](#) - Description

`trace(n)` One of the 8 [ComTraceColors](#) objects

`value` **(Long Integer)** - RGB color of the MemoryMarkers pen.

Convert the three RGB colors to an integer as follows:

```
RGB = R+(G*2^8)+(B*2^16)
```

To find the three RGB values from the [Display Colors dialog](#), click **Change Color**, then **Define Custom Color**.

Return Type Long

Default Varies for each trace.

Examples

```
R = 10
G = 10
B = 10
RGB = 10+(10*2^8)+(10*2^16)
trace1.MemoryMarkers = RGB 'Write
color = trace1.MemoryMarkers 'Read
```

C++ Syntax HRESULT get_MemoryMarkers(long* pVal);

HRESULT put_MemoryMarkers(long newVal);

Interface IComTraceColors

Last Modified:

7-Aug-2009 MX New topic

MessageText Property

Description	Returns text for the specified eventID
VB Syntax	<i>app</i> . MessageText , <i>eventID</i> , <i>message</i>
Variable	(Type) - Description
<i>app</i>	An Application (object)
<i>eventID</i>	(enum naEventID) Choose from the list in Working with the Analyzer's Events
<i>message</i>	(string) - variable to store the returned message
Return Type	String
Default	Not Applicable
Examples	<code>RFNA.MessageText naEventID_ARRANGE_WINDOW_EXCEED_CAPACITY, message</code>
C++ Syntax	HRESULT get_MessageText(tagNAEventID msgID, BSTR* message)
Interface	IApplication

MinimumFrequency Property

Description Sets and Returns the minimum frequency for the calibration standard.

VB Syntax *calstd.MinimumFrequency = value*

Variable [\(Type\)](#) - Description

calstd A CalStandard (**object**). Use calKit.[GetCalStandard](#) to get a handle to the standard.

value (**double**) -Minimum frequency in Hertz.

Return Type Double

Default Not Applicable

Examples `calstd.MinimumFrequency = 300e3 'Write`

`minFrequency = calstd.MinimumFrequency 'Read`

C++ Syntax HRESULT get_MinimumFrequency(double *pVal)
HRESULT put_MinimumFrequency(double newVal)

Interface ICalStandard

MinimumFrequency Property

Description Returns the minimum frequency of the remote PNA.

VB Syntax *value* = *cap*.**MinimumFrequency**

Variable [\(Type\)](#) - Description

value (Double) - Variable to store the returned minimum frequency of the PNA.

cap A [Capabilities](#) (object)

Return Type Double

Default Not Applicable

Examples `value = cap.MinimumFrequency 'Read`

C++ Syntax HRESULT get_MinimumFrequency(double *pVal)

Interface ICapabilities

MinimumFrequency Property

Description Sets and returns the minimum usable frequency specified for the power sensor.

VB Syntax *pwrSensor*.**MinimumFrequency** = *value*

Variable [\(Type\)](#) - Description

pwrSensor A [PowerSensor](#) (Object) or

A [PowerSensorAsReceiver](#) (Object)

value (**double**) -Frequency in Hertz.

Return Type Double

Default Device dependent

Examples `pwrSensor.MinimumFrequency = 300e3 'Write`

`MinFreq = pwrSensor).MinimumFrequency 'Read`

C++ Syntax HRESULT put_MinimumFrequency(double newVal);

HRESULT get_MinimumFrequency(double *pVal);

Interface IPowerSensor

IPowerSensorPMAR

Last Modified:

25-Aug-2009 Added PMAR

MinimumIFFrequency Property

Description	Returns the minimum allowed value for the IFFrequency Property
VB Syntax	<i>value</i> = <i>IfConfig</i> . MinimumIFFrequency
Variable	(Type) - Description
<i>value</i>	(double) Variable to store the returned minimum allowed frequency that can be applied to the IFFrequency Property .
<i>IfConfig</i>	IFConfiguration (object)
Return Type	Double
Default	Not Applicable
Examples	<code>val = App.ActiveChannel.IFConfiguration.MinimumIFFrequency</code> 'Read
C++ Syntax	HRESULT get_MinimumIFFrequency(double * pMinFreq);
Interface	IIFConfiguration3

Last Modified:

18-Jun-2007 MX New topic

MinimumNumberOfPoints Property

Description	Returns the minimum possible number of data points for a data trace.
VB Syntax	<i>value</i> = <i>cap</i> . MinimumNumberOfPoints
Variable	(Type) - Description
<i>value</i>	(Long) - Variable to store the returned minimum value for number of points.
<i>cap</i>	A Capabilities (object)
Return Type	Double
Default	Not Applicable
Examples	<pre>value = cap.MinimumNumberOfPoints 'Read</pre>
C++ Syntax	HRESULT get_MinimumNumberOfPoints(double * minimumNumberOfPoints);
Interface	ICapabilities

Last Modified:

24-Oct-2013	Changed return type
3-Dec-2007	MX New topic

MinimumReceiverStepAttenuator Property

Description	Returns a value indicating the minimum amount of receiver attenuation.
VB Syntax	<i>value</i> = <i>cap</i> . MinimumReceiverStepAttenuator (<i>n</i>)
Variable	(Type) - Description
<i>value</i>	(Double) - Variable to store the returned minimum value of receiver attenuation.
<i>cap</i>	A Capabilities (object)
<i>n</i>	(Long) - port number to query for minimum value of receiver attenuation
Return Type	Double
Default	Not Applicable
Examples	<pre>value = cap.MinimumReceiverStepAttenuator 'Read</pre>
C++ Syntax	<pre>HRESULTget_MinimumReceiverStepAttenuator(long portNumber, double * attenuation);</pre>
Interface	ICapabilities

MinimumSourceALCPower Property

Description	Returns a value indicating the minimum amount of source ALC power.
VB Syntax	<i>value</i> = <i>cap</i> . MinimumSourceALCPower (<i>n</i>)
Variable	(Type) - Description
<i>value</i>	(Double) - Variable to store the returned minimum value of source ALC power.
<i>cap</i>	A Capabilities (object)
<i>n</i>	(Long) - source number to query for the minimum value of source ALC power
Return Type	Double
Default	Not Applicable
Examples	<pre>value = cap.MinimumSourceALCPower 'Read</pre>
C++ Syntax	HRESULT get_MinimumSourceALCPower(long sourceNum, double * power);
Interface	ICapabilities

MixerCharacterizationFile Property

Description Set the filename of the S2P file used to characterize the calibration mixer

VB Syntax *smc.MixerCharacterizationFile* = *value*

Variable [\(Type\)](#) - Description

smc An [SMCType](#) (object)

value (String) Full path, file name, and extension of the mixer characterization file.

Return Type String

Default Not applicable

Example

```
SMC.MixerCharacterizationFile = "C:/Program  
Files/Agilent/Network Analyzer/Documents/default.S2P"
```

C++ Syntax HRESULT put_MixerCharacterizationFile(BSTR Value);

Interface SMCType5

Last Modified:

25-Mar-2010 MX New topic

Mode Property

Description	Sets the type of transform.
VB Syntax	<i>trans</i> .Mode = <i>value</i>
Variable	(Type) - Description
<i>trans</i>	A Transform (object)
<i>value</i>	(enum NATransformMode) - Choose from: 0 - naTransformBandpassImpulse 1 - naTransformLowpassImpulse 2 - naTransformLowpassStep
Return Type	NATransformMode
Default	0 - naTransformBandpassImpulse
Examples	<pre>trans.Mode = naTransformLowpassStep 'Write</pre> <pre>transmode = trans.Mode 'Read</pre>
C++ Syntax	HRESULT get_Mode(tagNATransformMode *pVal) HRESULT put_Mode(tagNATransformMode newVal)
Interface	ITransform

Mode Property

Description Sets and returns the stimulus mode for balanced measurements.

VB Syntax *balStim.Mode* = *value*

Variable [\(Type\)](#) - Description

balStim A [BalancedStimulus](#) (object)

value (Enum **NABALSTIMulus**) - Stimulus Mode. True modes are applicable only with [Opt 460 - iTMSA](#). When a True-Mode is selected, the Balanced port powers are automatically uncoupled.

Choose from:

0 - **naSEStim**: Single-Ended stimulus

1 - **naTMStim**: True-Mode stimulus

2 - **naFTMStim**: Forward only True-Mode stimulus

3 - **naRTMStim**: Reversed only True-Mode stimulus

Return Type Enum

Default 0 - **naSEStim**: Single-Ended stimulus

Examples

```
balStim.Mode = naTMStim 'Write
```

```
variable = balStim.Mode 'Read
```

[See an example iTMSA program](#)

C++ Syntax HRESULT get_Mode(tagNABALSTIMulus *eVal);

HRESULT put_Mode(tagNABALSTIMulus eVal);

Interface IBalancedStimulus

Last Modified:

15-May-2008 MX New topic

Move Property

Type topic text here.

Multiplier Property

Description Sets and returns the multiplier value to be used when coupling this range to the primary range. Learn more about multiplier value.

This setting is valid only if this range is [coupled](#) to the primary range.

VB Syntax `FOMRange.Multiplier = value`

Variable [\(Type\)](#) - Description

object An [FOMRange](#) (**object**)

value **(Double)** - Multiplier value.-(Unitless)

Return Type Double

Default 1

Examples `fomRange.Multiplier = 2 'Write`

`Mult = fomRange.Multiplier 'Read`

C++ Syntax HRESULT get_Multiplier(double *pVal)
HRESULT put_Multiplier(double pVal)

Interface IFOMRange

Name Property

Description Sets or returns the Name of the Cal Set.

VB Syntax *CalSet.Name* = *value*

Variable [\(Type\)](#) - Description

CalSet **(object)** - A [Cal Set](#) object

value **(string)** - Name of the Cal Set.

Return Type String

Default Not Applicable

Examples

```
Dim pna
set
pna=CreateObject("AgilentPNA835x.Application")

Dim calsets
set calsets=pna.getcalmanager.calsets

Dim c
for each c in calsets
wscript.echo c.name
'Changes the name of CalSet_1
if c.name="CalSet_1" then c.name="New"
next
```

C++ Syntax HRESULT get_Name(BSTR *name)

HRESULT put_Name(BSTR name);

Interface ICalSet4

Name (CalKit) Property

Description	Sets and Returns a name for the selected calibration kit.
VB Syntax	<i>calKit</i> .Name = <i>value</i>
Variable	(Type) - Description
<i>calKit</i>	A CalKit (object).
<i>value</i>	(string) -Calibration Kit name. Any string name, can include numerics, period, and spaces; any length (although the dialog box display is limited to about 30 characters).
Return Type	String
Default	Not Applicable
Examples	<pre>calKit.Name = "MyCalKit" 'Write KitName = calKit.Name 'Read</pre>
C++ Syntax	HRESULT get_Name(BSTR *pVal) HRESULT put_Name(BSTR newVal)
Interface	ICalKit

Name (PathConfig) Property

Description Returns the name of the current configuration only if NO individual element settings had been changed since selecting or storing a configuration. When element settings change, the path configuration name is cleared.

VB Syntax *name* = *pathConfig*.Name

Variable [\(Type\)](#) - Description

name **(String)** Variable to store the returned configuration name.

pathConfig A [PathConfiguration](#) **(object)**

Return Type String

Default 'default' - name of the default factory configuration

Examples `name=pathConf.Name`

C++ Syntax HRESULT get_Name(BSTR* ppName)

Interface IPathConfiguration

Last Modified:

14-Dec-2006 MX New topic

Name Property

Description Returns the name of the current element object

VB Syntax *name* = *pathElement*.Name

Variable [\(Type\)](#) - Description

name **(String)** Variable to store the returned element name.

pathElement A [PathElement](#) **(object)**

Return Type String

Default Not Applicable

Examples `name=pathElement.Name`

C++ Syntax HRESULT get_Name(BSTR* ppName)

Interface IPathElement

Last Modified:

15-Dec-2006 MX New topic

Name (ExternalDevice) Property

Description Sets and returns the name of the External Device.

VB Syntax *extDev*.Name = *value*

Variable [\(Type\)](#) - Description

extDev A [ExternalDevice Object](#) (object).

value (string) - External Device name. Any string name limited to alpha-numeric characters.

Return Type String

Default Device<n>

Examples

```
extDev.Name = "MySource" 'Write
extDevName = extDev.Name 'Read

See example program to configure PMAR device
See example program to configure External Source
```

C++ Syntax HRESULT get_Name(BSTR *pVal)

HRESULT put_Name(BSTR newVal)

Interface IExternalDevice

Last Modified:

31-Jul-2009 MX New topic

Name Property

Description Returns the name of this FOM range object.

VB Syntax *value* = *FOMRange*.Name

Variable [\(Type\)](#) - Description

value **(string)** - Variable to store the returned range name.

FOMRange An [FOMRange](#) **(object)**

Return Type String

Default Not Applicable

Examples `rname = fomRange.Name 'Read`

C++ Syntax HRESULT get_Name(BSTR *pRName)

Interface IFOMRange

Last Modified:

8-Mar-2007 Major Modifications

Name (Measurement) Property

Description Sets (or returns) the Name of the measurement. Measurement names must be unique among the set of measurements. Measurement names cannot be an empty string.

Note: This is the same name as trace.Name; when one changes, the other changes.

VB Syntax `meas.Name = value`

Variable [\(Type\)](#) - Description

meas A Measurement (**object**)

value (**string**) - A user defined name of the measurement

Return Type String

Default "CH1_S11_1" - name of the default measurement

Examples

```
meas.Name = "Filter BPass" 'Write
```

```
MName = meas.Name 'Read
```

C++ Syntax HRESULT get_Name(BSTR *pVal)
HRESULT put_Name(BSTR newVal)

Interface IMeasurement

Name Property

Description	Sets and returns the name of the Power Sensor (object) to be used as part of a Guided Power Calibration.
VB Syntax	<i>pSensor</i> .Name = <i>value</i>
Variable	(Type) - Description
<i>pSensor</i>	A GuidedCalibrationPowerSensor (object)
<i>value</i>	(string) - Name of the power sensor. The power sensor must be already configured as a PMAR device using this name. Learn how to remotely configure a PMAR device.
Return Type	String
Default	Not Applicable
Examples	<pre>Sensor.Name = "26GHzSesnsor" 'write</pre> <pre>value = pSensor.Name 'Read</pre> <p>See example program</p>
C++ Syntax	HRESULT get_Name(BSTR *pPsensorName) HRESULT put_Name(BSTR pPsensorName)
Interface	IGuidedCalibrationPowerSensor

Last Modified:

8-Feb-2011 New topic

Name (trace) Property

Description Sets or returns the name of the Trace. Use the trace name to identify the trace and refer to the trace in the collection.

Note: This is the same name as `meas.Name`; when one changes, the other changes.

VB Syntax `trac.Name = value`

Variable [\(Type\)](#) - Description

`trac` A Trace (**object**)

`value` (**String**) Trace name

Return Type String

Default "CH1_S11_1" - name of the default measurement

Examples `trace.Name = "myTrace" 'Write`

`traceName = Name.Trace 'Read`

C++ Syntax HRESULT put_Name(BSTR name)
HRESULT get_Name(BSTR *name)

Interface ITrace

NetworkFilename Property

Description	Specifies the S2P filename to embed or de-embed on the input or output of your mixer measurement.
VB Syntax	<i>object</i> .NetworkFilename(<i>n</i>) = <i>filename</i>
Variable	(Type) - Description
<i>object</i>	SMCType (object) or VMCType (object)
<i>n</i>	(Integer) Apply network to input or output of mixer. Choose from: 1 - Input of mixer 2 - Output of mixer
<i>filename</i>	(String) Filename of the S2P used for embedding or de-embedding. Use the full path name, file name, and .S2P suffix, enclosed in quotes.
Return Type	String
Default	Not Applicable
Examples	<pre>VMC.Filename(2) = "c:\users\public\network analyzer\documents\WaveguideAdapt.S2P"</pre>
C++ Syntax	HRESULT put_NetworkFilename(short networkNum, BSTR filename); HRESULT get_NetworkFilename(short networkNum, BSTR *filename);
Interface	SMCType2 VMCType2

Last Modified:

29-Feb-2008 Several edits

NetworkMode Property

Description	Allows you to embed (add) or de-embed (remove) circuit network effects on the input and output of your mixer measurement.
VB Syntax	<i>object</i> .NetworkMode(<i>n</i>) = <i>value</i>
Variable	(Type) - Description
<i>object</i>	SMCType (object) or VMCType (object)
<i>n</i>	(Integer) Apply network to input or output of mixer. Choose from: 1 - Input of mixer 2 - Output of mixer
<i>value</i>	(Enum as ENetworkMode) Choose from: NO_NETWORK Do nothing with effects of S2P file EMBED_NETWORK - Add effects of S2P file from the measurement results. DEEMBED_NETWORK - Remove effects of S2P file from the measurement results.
Return Type	Enum
Default	NO_NETWORK
Examples	<pre>VMC.NetworkMode = EMBED_NETWORK</pre>
C++ Syntax	HRESULT put_NetworkMode(short networkNum, enum ENetworkMode networkMode); HRESULT get_NetworkMode(short networkNum, enum ENetworkMode *networkMode);
Interface	SMCType2 VMCType2

NetworkPortMapA Property

Description Read the port mapping of “in A” port for a 4-port SNP file to be embedded.

Use [NetworkPortMap Method](#) to set the port map for all four ports.

VB Syntax `value = fixture.NetworkPortMapA (network)`

Variable (Type) - Description

value (Long) Variable to store the returned value.

fixture A [Fixturing \(object\)](#)

network (Integer) Network position. Choose from **1** or **2**.

Default 1

Examples `value = fixture.NetworkPortMapA(1)`

C++ Syntax HRESULT get_NetworkPortMapA(short* Net);

Interface IFixturing6

Last Modified:

10-Jan-2012 Major edits

16-Nov-2010 MX New topic

NetworkPortMapB Property

Description Read the port mapping of “in B” port for a 4-port SNP file to be embedded.

Use [NetworkPortMap Method](#) to set the port map for all four ports.

VB Syntax *value* = *fixture*.**NetworkPortMapB** (*network*)

Variable **(Type) - Description**

value (Long) Variable to store the returned value.

fixture A [Fixturing \(object\)](#)

network (Integer) Network position. Choose from **1** or **2**.

Default 2

Examples `value = fixture.NetworkPortMapB(1)`

C++ Syntax HRESULT get_NetworkPortMapB(short* Net);

Interface IFixturing6

Last Modified:

10-Jan-2012 Major edits

16-Nov-2010 MX New topic

NetworkPortMapC Property

Description Read the port mapping of “out A” port for a 4-port SNP file to be embedded.

Use [NetworkPortMap Method](#) to set the port map for all four ports.

VB Syntax *value* = *fixture*.**NetworkPortMapC** (*network*)

Variable [\(Type\)](#) - **Description**

value (Long) Variable to store the returned value.

fixture A [Fixturing \(object\)](#)

network (Integer) Network position. Choose from **1** or **2**.

Default 3

Examples `value = fixture.NetworkPortMapC(1)`

C++ Syntax HRESULT get_NetworkPortMapC(short* Net);

Interface IFixturing6

Last Modified:

10-Jan-2012 Major edits

16-Nov-2010 MX New topic

NetworkPortMapD Property

Description Read the port mapping of “out B” port for a 4-port SNP file to be embedded.

Use [NetworkPortMap Method](#) to set the port map for all four ports.

VB Syntax *value* = *fixture*.**NetworkPortMapD** (*network*)

Variable [\(Type\)](#) - **Description**

value (Long) Variable to store the returned value.

fixture A [Fixturing \(object\)](#)

network (Integer) Network position. Choose from **1** or **2**.

Default 4

Examples `value = fixture.NetworkPortMapD(1)`

C++ Syntax HRESULT get_NetworkPortMapD(short* Net);

Interface IFixturing6

Last Modified:

10-Jan-2012 Major edits

16-Nov-2010 MX New topic

NoiseAverageFactor Property

Description Sets and reads the averaging of the noise receiver.

VB Syntax *noise*.NoiseAverageFactor = *value*

Variable [\(Type\)](#) - Description

noise A [NoiseFigure](#) (object)

value (**long integer**) - Averaging value. Choose a number between 1 and 99.

Return Type Long Integer

Default 1

Examples `noise.NoiseAverageFactor = 10 'Write`

`AvgNoise = noise.NoiseAverageFactor 'Read`

C++ Syntax HRESULT get_NoiseAverageFactor(long* pVal)

HRESULT put_NoiseAverageFactor(long newVal)

Interface INoiseFigure

Last Modified:

29-May-2007 MN New topic

NoiseAverageState Property

Description	Turns Noise Averaging ON and OFF.
VB Syntax	<i>noise</i> .NoiseAverageState = <i>value</i>
Variable	(Type) - Description
<i>noise</i>	A NoiseFigure (object)
<i>value</i>	(boolean) - Averaging state.
	False - Turns Noise Averaging OFF
	True - Turns Noise Averaging ON
Return Type	Boolean
Default	False
Examples	<pre>noise.NoiseAverageState = OFF 'Write</pre>
	<pre>NoiseAvgState = noise.NoiseAverageState 'Read</pre>
C++ Syntax	HRESULT get_NoiseAverageState(VARIANT_BOOL* on); HRESULT put_NoiseAverageState(VARIANT_BOOL on);
Interface	INoiseFigure

Last Modified:

21-Jun-2007 MX New topic

NoiseBandwidth Property

Description Set the bandwidth of the noise receiver.

VB Syntax `noise.NoiseBandwidth = value`

Variable [\(Type\)](#) - Description

noise A [NoiseFigure](#) (object)

value (**double**) Bandwidth value.

For [NoiseReceiver](#) = **naNoiseReceiver** (Opt 029) choose from: 800 KHz, 2 MHz, 4 MHz, 8 MHz, or 24 MHz or the numerical equivalent, such as 8e6 and so forth. **NOTE:** The [Receiver Characterization Method](#) = "Power Meter" is NOT allowed when the Noise Bandwidth is 8 MHz or 24 MHz.

For NoiseReceiver = **naStandardReceiver** (Opt 028) choose from: 720 kHz or 1.2 MHz

If the value does not match one of these, it is rounded up to the next valid bandwidth value.

Return Type Double

Default 4 MHz for **naNoiseReceiver**

1.2 MHz for **naStandardReceiver**

Examples `noise.NoiseBandwidth = 2E6 'Write`

`NoiseBW = noise.NoiseBandwidth 'Read`

C++ Syntax `HRESULT get_NoiseBandwidth(double *pVal);`

`HRESULT put_NoiseBandwidth(double newVal);`

Interface INoiseFigure

Last Modified:

21-May-2012 Edited for RcvrCharMethod
21-Apr-2010 Added noise receiver options
20-Sep-2007 MX New topic

NoiseReceiver Property

Description Sets and returns the receiver to use for noise measurements.

VB Syntax *noise.NoiseReceiver* = *value*

Variable [\(Type\)](#) - Description

noise A [NoiseFigure](#) (object)

value (Enum as **NaNNoiseReceiverMode**) Noise receiver. Choose from:

0 - naStandardReceiver The standard PNA receiver. (Opt 028 or 029)

1 - naNoiseReceiver The noise receiver. (Opt 029 ONLY)

Return Type Enum

Default **1 - naNoiseReceiver**

Examples `noise.NoiseReceiver = naNoiseReceiver 'Write`

`NoiseRec = noise.NoiseReceiver 'Read`

C++ Syntax HRESULT get_NoiseReceiver(tagNoiseReceiverMode *pVal);

HRESULT put_NoiseReceiver(tagNoiseReceiverMode newVal);

Interface INoiseFigure5

Last Modified:

11-Jul-2011 Edit options

2-Mar-2010 MX New topic

NoiseReceiverSweepTime Property

Description Returns the APPROXIMATE time the channel will take to make one noise receiver sweep given the current setup. This, along with the sweep time for a standard receiver measurement and the following calculations, can tell you how long a “single” sweep would take so that you can set an appropriate "timeout" in your program.

Use [Sweep Time Property](#) to perform the standard sweep time query, shown as **SSwpTime** below.

To calculate the total sweep time:

Noise Figure on amplifiers (Vector Correction ON):

- $2 * \text{SSwpTime} + X * \text{NoiseReceiverSweepTime}$
- Where X = the number of noise receiver impedance state sweeps. (Default is 4).

Noise Figure on converters (NFX) correction on - increased number of sweeps due to extra mixer sweeps and source pulling:

- $4 * \text{SSwpTime} + X * \text{NoiseReceiverSweepTime}$ (without source pulling)
- $8 * \text{SSwpTime} + X * \text{NoiseReceiverSweepTime}$ (with source pulling)
- Where X = the number of noise receiver impedance state sweeps. (Default is 4).

Note: The number of sweeps to perform a noise measurement is annotated at the bottom of the Noise Figure screen.

VB Syntax $\text{swpTime} = \text{NF.NoiseReceiverSweepTime}$

Variable [\(Type\)](#) - **Description**

swpTime Variable to store the returned sweep time value (in seconds).

NF A [NoiseFigure](#) (**object**)

Return Type Double

Default Not Applicable

Examples $\text{swpTime} = \text{NF.NoiseReceiverSweepTime}()$

C++ Syntax HRESULT get_NoiseReceiverSweepTime (Double* SwpTime);

Interface INoiseFigure3

Last Modified:

28-Oct-2009 MX New topic

NoiseGain Property

Description Sets and reads the gain state of the noise receiver. This setting is NOT used when [NoiseReceiver](#) = naStandardReceiver (Opt 028)

VB Syntax `noise.NoiseGain = value`

Variable [\(Type\)](#) - Description

noise A [NoiseFigure](#) (**object**)

value (**long integer**) - Gain value. Choose from:

0 Low Gain select if the gain of your DUT is relatively high (>35 dB).

15 Medium Gain select if the gain of your DUT is about average (20 dB to 45 dB).

30 High Gain ..select if the gain of your DUT is relatively low (<30 dB).

If the value does not match one of these, it is rounded up to the next legal value.

[Learn more about Noise Receiver Gain setting.](#)

Return Type Long Integer

Default 30

Examples

```
noise.NoiseGain = 30 'Write
```

```
GainNoise = noise.NoiseGain 'Read
```

C++ Syntax HRESULT get_NoiseGain(long* pVal)

HRESULT put_NoiseGain(long newVal)

Interface INoiseFigure

Last Modified:

29-May-2007 MN New topic

NoiseSourceCalKitType Property

Description Set and read the Cal Kit that will be used for the Noise Source adapter.

An adapter is always necessary to connect a 346C Noise Source to the PNA port 2. Select a Cal Kit that is the same type and gender as the noise source connector.

If the Noise Source mates directly to PNA port 2, then set this type to "None".

VB Syntax `noise.NoiseSourceCalKitType = value`

Variable [\(Type\)](#) - Description

noise A [NoiseCal](#) (**object**)

value (**string**) Cal Kit type. To read possible cal kit strings for the adapter:

1. Change the port connector type to that of the noise source using: [ConnectorType](#)
2. Then read the possible cal kit strings for that port using: [CompatibleCalKits](#)

Return Type String

Default Not applicable

Examples

```
noise.NoiseSourceCalKitType = "N4691-60004 ECAL" 'Write
calkit = noise.NoiseSourceCalKitType 'Read
```

C++ Syntax HRESULT get_NoiseSourceCalKitType(BSTR* pValue)
HRESULT put_NoiseSourceCalKitType(BSTR pNewValue)

Interface INoiseCal

Last Modified:

29-May-2007 MN New topic

NoiseSourceCold Property

Description Sets and returns the temperature of the noise source connector.

VB Syntax *noise.NoiseSourceCold* = *value*

Variable [\(Type\)](#) - Description

noise A [NoiseCal](#) (object)

value (**double**) Noise source temperature in Kelvin.

Return Type Double

Default Not Applicable

Examples `noise.NoiseSourceCold = 295 'Write`

`temp = noise.NoiseSourceCold 'Read`

C++ Syntax HRESULT get_NoiseSourceCold(double* pTemp)

HRESULT put_NoiseSourceCold(double pNewTemp)

Interface INoiseCal

Last Modified:

21-Jun-2007 MX New topic

NoiseSourceConnectorType Property

Description Set and read the Noise Source connector type and gender.

The Agilent 346C has an "APC 3.5 male" connector.

VB Syntax *noise.NoiseSourceConnectorType* = *value*

Variable [\(Type\)](#) - Description

noise A [NoiseCal](#) (object)

value (**string**) Connector type. Use [ValidConnectorType](#) to return a list of valid connector types.

Return Type String

Default Not applicable

Examples

```
noise.NoiseSourceConnectorType = "APC 3.5 male" 'Write  
connector = noise.NoiseSourceConnectorType 'Read
```

C++ Syntax HRESULT get_NoiseSourceConnectorType(BSTR* pConnectorType)
HRESULT put_NoiseSourceConnectorType(BSTR pConnectorType)

Interface INoiseCal

Last Modified:

29-May-2007 MN New topic

NoiseSourceState Property

Description Sets and reads the noise source ([28V](#)) ON and OFF.

VB Syntax `app.NoiseSourceState = state`

Variable [\(Type\)](#) - Description

app An [Application](#) (**object**)

state (**boolean**)

False (0) - Turns Noise Source OFF

True (1) - Turns Noise Source ON

Return Type Boolean

Default For PNA models with a Noise Figure option (028/029/H29), the 28V line is ON at application start and after a preset. The ON/OFF state is also available from a PNA softkey menu.

For PNA models WITHOUT a Noise Figure option (028/029/H29), the 28V line is OFF at application start and it's state is not affected by a preset. The ON/OFF state is NOT available from a PNA softkey menu.

Examples `app.NoiseSourceState = True 'Write`

`coupl = app.NoiseSourceState 'Read`

C++ Syntax HRESULT put_NoiseSourceState(VARIANT_BOOL bState)
HRESULT get_NoiseSourceState(VARIANT_BOOL *bState)

Interface IApplication13

Last Modified:

3-Nov-2010 Available for PNA models without a NF option.

29-May-2007 MN New topic

NoiseTuner Property

Description Sets and returns the noise tuner identifier, which is an ECal model and serial number string.

VB Syntax `noise.NoiseTuner = value`

Variable [\(Type\)](#) - Description

noise A [NoiseFigure](#) (**object**)

value (**string**) Noise Tuner. Return the connected ECal identifiers by sending GetCalKitTypeString and passing the module number.

Return Type String

Default Not Applicable

Examples

```
noise.NoiseTuner = "N4691-60004 ECal 02822" 'Write
noiseT = noise.NoiseTuner 'Read
```

C++ Syntax HRESULT get_NoiseTuner(BSTR* pValue)
HRESULT put_NoiseTuner(BSTR pNewValue)

Interface INoiseFigure

Last Modified:

18-Jun-2007 MN New topic

NoiseTunerIn Property

Description	Sets and returns the port identifier of the ECal noise tuner that is connected to the PNA Source.
VB Syntax	<i>noise.NoiseTunerIn = value</i>
Variable	(Type) - Description
<i>noise</i>	A NoiseFigure (object)
<i>value</i>	(string) Noise Tuner port identifier that is connected to the PNA source, as it is labeled on the ECal module. For example, for 2-port ECal modules, choose either "A" or "B".
Return Type	String
Default	Not Applicable
Examples	<pre>noise.NoiseTunerIn = "A" 'Write EcalPort = noise.NoiseTunerIn 'Read</pre>
C++ Syntax	HRESULT get_NoiseTunerIn(BSTR* pValue) HRESULT put_NoiseTunerIn(BSTR pNewValue)
Interface	INoiseFigure

Last Modified:

29-May-2007 MN New topic

NoiseTunerOut Property

Description Sets and returns the port identifier of the ECal noise tuner that is connected to the DUT.

VB Syntax `noise.NoiseTunerOut = value`

Variable [\(Type\) - Description](#)

noise A [NoiseFigure](#) (object)

value (**string**) Noise Tuner port identifier that is connected to the DUT, as it is labeled on the ECal module. For example, for 2-port ECal modules, choose either "A" or "B".

Return Type String

Default Not Applicable

Examples

```
noise.NoiseTunerOut = "A" 'Write
EcalPort = noise.NoiseTunerOut 'Read
```

C++ Syntax HRESULT get_NoiseTunerOut(BSTR* pValue)
HRESULT put_NoiseTunerOut(BSTR pNewValue)

Interface INoiseFigure

Last Modified:

29-May-2007 MN New topic

NominalIncidentPowerState Property

Description	Toggles the Nominal Incident Power setting ON and OFF. This setting is ONLY to be used with SMC measurements, not VMC. Learn more about Nominal Incident Power.
VB Syntax	<i>mixer</i> . NominalIncidentPowerState = <i>bool</i>
Variable	(Type) - Description
<i>mixer</i>	A Mixer (object)
<i>bool</i>	(boolean) - Nominal Incident Power State. Choose from: 1 -(True) Turn nominal incident power ON 0 -(False) Turn nominal incident power OFF
Return Type	Boolean
Default	0 -(False)
Examples	<pre>mixer.NominalIncidentPowerState = True 'sets NominalIncidentPowerState to ON</pre>
C++ Syntax	HRESULT get_NominalIncidentPowerState(VARIANT_BOOL *pVal) HRESULT put_NominalIncidentPowerState(VARIANT_BOOL val);
Interface	IMixer4

NormalizePoint Property

Description	Sets and returns the sweep data point around which to perform broadband and precise tuning.
VB Syntax	<i>obj</i> . NormalizePoint = <i>value</i>
Variable	(Type) - Description
<i>obj</i>	An EmbeddedLO (object) or A ConverterEmbeddedLO (object)
<i>value</i>	(Long) Mixer Sweep data point. Choose a data point number between 1 and the max number of data points in the sweep that has the least amount of expected noise.
Return Type	(Long)
Default	Center point in the sweep span
Examples	<pre>embedLO.NormalizePoint = 101 'write</pre> <pre>value = embedLO.NormalizePoint 'read</pre>
C++ Syntax	HRESULT get_NormalizePoint(long *point); HRESULT put_NormalizePoint(long point);
Interface	IEmbeddedLO

Last Modified:

12-Aug-2009 Added ConvEmbedLO object

18-Apr-2007 MX New topic

NormalizePoint Property

Description Sets or returns the data point used for normalizing the phase measurement.

VB Syntax *obj.NormalizePoint = value*

Variable [\(Type\)](#) - Description

obj A [Mixer Interface](#) pointer to the [Measurement](#) (object)

A [Converter](#) (Object)

value **(Long)** - Normalization data point. Choose a data point number between 1 and the max number of data points in the sweep that has the least amount of expected noise.

Return Type Long Integer

Default Center point in the sweep

Examples `mixer.NormalizePoint = 101`

C++ Syntax HRESULT get_NormalizePoint(Long * val);

HRESULT put_NormalizePoint(Long val);

Interface IMixer13

Last Modified:

25-Mar-2010 MX New topic

Number (Measurement) Property

Description Returns the Number of the measurement. Measurement numbers are assigned internally.

Note: Measurement numbers are NOT the same as their number in the Measurements collection. Measurement number is used to identify the measurement associated with an event.

This property is used to identify measurements when events occur through the [OnMeasurementEvent](#) callback. For example:

```
OnMeasurementEvent (naEventId_MSG_LIMIT_FAILED, 3)
```

VB Syntax `measNum = meas.Number`

Variable [\(Type\)](#) - Description

`measNum` **(long)** - variable to store the measurement number

`meas` A Measurement **(object)**

Return Type Long Integer

Default "1" - number of the default measurement

Examples `measNum = meas.Number`

C++ Syntax HRESULT get_Number(long *MeasurementNumber)

Interface IMeasurement

Number Property

Description	Returns the number of the Auxiliary Trigger connector pair currently being used with the instance of the AuxTrigger object. Set the trigger pair with the AuxTrig object.
VB Syntax	<i>value</i> = <i>auxTrig</i> . Number
Variable	(Type) - Description
<i>auxTrig</i>	An AuxTrigger (object)
<i>value</i>	(Long Integer) - Connector pair. PNA-X returns 1 or 2. All other models that do not have the Aux trigger connector returns 1.
Return Type	Single
Default	Not Applicable
Exaamples	<code>value = auxTrig.Number 'Read the value</code>
C++ Syntax	HRESULT get_Number(long *auxID);
Interface	IAuxTrigger

Last Modified:

7-Apr-2009	Updated to "all other models.."
14-Dec-2006	MX New topic

NumberOfFrequencyPoints Property

Description Set and read the number of frequency points for a Gain Compression channel. Applies to all acquisition modes.

VB Syntax `gca.NumberOfFrequencyPoints = value`

Variable [\(Type\)](#) - Description

gca A [GainCompression](#) (object)

value (**integer**) - Frequency points. Do not exceed the max number of points. [Learn more.](#)

Return Type Integer

Default 201

Examples `gca.NumberOfFrequencyPoints = 101 'Write`

`freqPts = gca.NumberOfFrequencyPoints 'Read`

C++ Syntax HRESULT get_NumberOfFrequencyPoints(int* pVal)

HRESULT put_NumberOfFrequencyPoints(int newVal)

Interface IGainCompression

Last Modified:

11-Sep-2007 MX New topic

NumberOfPoints Property

Description Sets or returns the Number of Points of the channel.

Sets or returns the Number of Points of the segment.

See Also

- [Measurement2 Interface](#) to learn how this method differs from [meas.NumberofPoints](#)
- [Gain Compression Number of Points](#).
- [Swept IMD limitations](#)

VB Syntax *object*.NumberOfPoints = *value*

Variable [\(Type\)](#) - Description

object Channel (**object**)

or

CalSet (**object**) - Read-only property

value (**long**) - Number of Points.

For channel, choose any number from **1** to the [PNA max number of points](#).

For segment, the total number of points in all segments cannot exceed the PNA maximum. A segment can have as few as 1 point.

Return Type Long Integer

Default 201 for channel

21 for segment

Examples

```
chan.NumberOfPoints = 201 'sets the number of points for all  
measurements in the channel. -Write
```

```
numofpts = chan.NumberOfPoints 'Read
```

C++ Syntax HRESULT get_NumberOfPoints(long *pVal)
HRESULT put_NumberOfPoints(long newVal)

Interface IChannel
ISegment
|CalSet3

Last Modified:

5-Aug-2008	Fixed bad link
21-Jun-2007	Increased max

Read-only

NumberOfPoints Property

Description	Returns the number of data points of the measurement. To understand how this property is useful, see IMeasurement2 Interface .
VB Syntax	<i>value</i> = <i>meas</i> . NumberOfPoints
Variable	(Type) - Description
<i>value</i>	(Long) - variable to store the returned value
<i>meas</i>	A Measurement (object)
Return Type	Long Integer
Default	Not Applicable
Examples	<pre>Print meas.NumberOfPoints 'prints the number of data points</pre>
C++ Syntax	HRESULT get_NumberOfPoints(long *pVal);
Interface	IMeasurement2

Read-only

NumberOfPorts Property

Description Returns the number of hardware source ports on the PNA.

VB Syntax *value* = *app*.NumberOfPorts

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

value (long integer) - variable to contain the returned value

Return Type (long integer)

Default Not Applicable

Examples `iNumPorts = app.NumberOfPorts`

C++ Syntax HRESULT NumberOfPorts(long* NumPorts)

Interface IApplication

NumberOfPorts Property

Description	Returns the number of ports on the specified testset. Returns 0 if no test set is connected.
VB Syntax	<i>value</i> = <i>tset</i> . NumberOfPorts
Variable	(Type) - Description
<i>value</i>	(Long) variable to store the returned information.
<i>tset</i>	A TestsetControl object.
	OR
	An E5091Testset object.
Return Type	Long Integer
Default	Not Applicable
Examples	<pre>value = testset1.NumberOfPorts</pre> See E5091A Example Program See External Testset Program
C++ Syntax	HRESULT get_NumberOfPorts(long *numberOfPorts);
Interface	ITestsetControl IE5091Testset

NumberOfPowerPoints Property

Description Set and read the number of data points in each power sweep. Applies ONLY to 2D [acquisition modes](#).

VB Syntax `gca.NumberOfPowerPoints = value`

Variable [\(Type\)](#) - Description

gca A [GainCompression](#) (object)

value (**integer**) - Power points. Do not exceed the max number of points.

For 2D sweeps, total = frequency x power. Max = 20,001

For Smart sweep, total = frequency. Max = 10,000.

Return Type Integer

Default 26

Examples

```
gca.NumberOfPowerPoints = 31 'Write
```

```
pwrPts = gca.NumberOfPowerPoints 'Read
```

C++ Syntax HRESULT get_NumberOfPowerPoints(int* pVal)
HRESULT put_NumberOfPowerPoints(int newVal)

Interface IGainCompression

Last Modified:

11-Sep-2007 MX New topic

NumberOfSweeps Property

Description Returns the number of tuning sweeps used for the latest embedded LO measurement.

VB Syntax *value* = *embedLODiag*.NumberOfSweeps

Variable [\(Type\)](#) - Description

value **(Long)** Variable to store the returned data.

embedLODiag An [EmbeddedLODiagnostic](#) **(object)**

Return Type **(Long)**

Default Not Applicable

Examples `data= embedLODiag.NumberOfSweeps 'read`

C++ Syntax HRESULT get_NumberOfSweeps(long * numSweeps);

Interface IEmbeddedLODiagnostic

Last Modified:

12-Apr-2007 MX New topic

Offset Property

Description	Sets and returns the offset value to be used when coupling this range to the primary range. This setting is valid only if the specified range is coupled to the primary range.
VB Syntax	<i>FOMRange</i> . Offset = <i>value</i>
Variable	(Type) - Description
<i>object</i>	An FOMRange (object)
<i>value</i>	(Double) - Offset value.-(Unitless)
Return Type	Double
Default	0
Examples	<pre>fomRange.Offset = 1e9 'Write</pre> <pre>Offs = fomRange.Offset 'Read</pre>
C++ Syntax	HRESULT get_Offset(double *pVal) HRESULT put_Offset(double pVal)
Interface	IFOMRange

Last Modified:

8-Mar-2007 Modified links.

OffsetReceiverAttenuator Property

Description Set and return whether to offset the reference receiver by the amount of receiver attenuation. [Learn more.](#)

This setting remains until changed or until the hard drive is changed or reformatted.

VB Syntax `pref.OffsetReceiverAttenuator = value`

Variable (Type) - Description

pref A [Preferences](#) (object)

value (Boolean) - Choose from:

False Do NOT offset the test port receivers.

True Offset the test port receivers.

Return Type Boolean

Default **True** PNA-X models

False E836xB and PNA-L models

Examples

```
pref.OffsetReceiverAttenuator = 1 'Write
```

```
Rcvroffset = pref.OffsetReceiverAttenuator 'Read
```

C++ Syntax HRESULT get_OffsetReceiverAttenuator(VARIANT_BOOL * val);

HRESULT put_OffsetReceiverAttenuator(VARIANT_BOOL val);

Interface IPreferences6

Last Modified:

14-Jan-2007 MX New topic

OffsetSourceAttenuator Property

Description Set and return whether to mathematically offset the reference receivers by the amount of source attenuation. [Learn more.](#)

This setting remains until changed or until the hard drive is changed or reformatted.

VB Syntax `pref.OffsetSourceAttenuator = value`

Variable [\(Type\)](#) - Description

pref A [Preferences](#) (object)

value **(Boolean)** - Choose from:

False Do NOT offset the reference receivers.

True Offset the reference receivers.

Return Type Boolean

Default **True** PNA-X models

False E836xB and PNA-L models

Examples

```
pref.OffsetSourceAttenuator = 1 'Write
```

```
Rcvroffset = pref.OffsetSourceAttenuator 'Read
```

C++ Syntax HRESULT get_OffsetSourceAttenuator(VARIANT_BOOL * val);

HRESULT put_OffsetSourceAttenuator(VARIANT_BOOL val);

Interface IPreferences6

Last Modified:

14-Jan-2007 MX New topic

OmitIsolation Property - Superseded

Description	This command is replaced with SetIsolationPaths and GetIsolationPaths . Sets and returns whether Isolation portion of the calibration will be performed or not.
VB Syntax	<i>obj.OmitIsolation</i> = <i>bool</i>
Variable	(Type) - Description
	<i>obj</i> SMCType (object)
	or
	VMCType (object)
	<i>bool</i> (Boolean)
	True - Isolation is NOT performed
	False - Isolation is performed
Return Type	Boolean
Default	True
Examples	<pre>value = SMC.OmitIsolation</pre>
C++ Syntax	HRESULT put_OmitIsolation (VARIANT_BOOL bState) HRESULT get_OmitIsolation (VARIANT_BOOL *bState)
Interface	SMCType VMCType

OneMarkerReadoutPerTrace Property - Superseded

Description Either show marker readout of only the active trace or all of the traces simultaneously.

Note: This method is replaced by [MarkerReadoutsPerTrace Property](#)

VB Syntax `win.OneReadoutPerTrace = state`

Variable [\(Type\)](#) - Description

win A [NAWindow](#) (object)

value **(boolean)**

True - Shows the readout of only the active marker for each trace.

False - Shows up to 5 marker readouts per trace, up to 20 total readouts.

Return Type Boolean

Default False

Examples

```
win.OneMarkerReadoutPerTrace = True 'Write
```

```
State = app.ActiveNAWindow.OneMarkerReadoutPerTrace 'Read
```

C++ Syntax HRESULT get_OneMarkerReadoutPerTrace(VARIANT_BOOL *pVal)

HRESULT put_OneMarkerReadoutPerTrace(VARIANT_BOOL newVal)

Interface INAWindow

Options Property

Description Returns a string identifying the analyzer option configuration.

VB Syntax *value* = *app.Options*

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

value (string) - variable to contain the returned string

Return Type String

Default Not Applicable

Examples `availOptions = app.Options`

C++ Syntax HRESULT Options(BSTR* OptionString)

Interface IApplication

OrientECALModule Property

Description Specifies if the PNA should perform orientation of the ECal module during calibration. Orientation is a technique by which the PNA automatically determines which ports of the module are connected to which ports of the PNA. Orientation begins to fail at very low power levels or if there is much attenuation in the path between the PNA and the ECal module.

Note: For guided calibrations where Orient is OFF and the same ECal module is used in more than one Connection Step, you are not allowed to specify how the ECal module is connected. Instead, the PNA determines the orientation. The PNA does not verify that you made the connection properly.

This setting remains until the PNA is restarted or this command is sent again.

This command, and [ECALPortMapEx](#), can be used to perform ECal using the [Guided Calibration](#) interface.

VB Syntax `cal.OrientECALModule = value`

Variable [\(Type\)](#) - **Description**

cal A [Calibrator](#) (object)

value (boolean)

False – [DoECAL1PortEX](#) and [DoECAL2PortEx](#) methods will use the value of the [ECALPortMapEx](#) property to determine the port connections.

True - [DoECAL1PortEX](#) and [DoECAL2PortEx](#) methods will use auto Orientation technique to determine port connections.

Return Type Boolean

Default True

Examples Dim cal As Calibrator

Dim bOrient As Boolean

Set cal = PNAapp.ActiveChannel.Calibrator

cal.OrientECALModule = False 'Write

bOrient = cal.OrientECALModule 'Read

C++ Syntax HRESULT put_OrientECALModule(VARIANT_BOOL bOrient);
HRESULT get_OrientECALModule(VARIANT_BOOL *bOrient);
Interface ICalibrator3

Last Modified:

7-May-2007 Added note about Guided Cal.

OutputChannel Property

Description Sets and returns the output channel of the pulse generator.

VB Syntax *extPulseGen*.**OutputChannel** = *value*

Variable [\(Type\)](#) - [Description](#)

extPulseGen An [ExternalPulseGenerator](#) (**object**)

value (**Long**) Pulse Generator output port. Choose from 1 or 2.

Return Type Long Integer

Default 1

Examples

```
extPulseGen.OutputChannel = 2 'Write
port = extPulseGen.OutputChannel 'Read
```

C++ Syntax HRESULT get_OutputChannel (long *pValue)
HRESULT put_OutputChannel (long newVal)

Interface IExternalPulseGenerator

Last Modified:

15-Feb-2012 New topic

OutputFixedFrequency Property

Description Sets or returns the mixer output fixed frequency value.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

VB Syntax *obj*.OutputFixedFrequency = *value*

Variable [\(Type\)](#) - Description

obj A [Mixer Interface](#) pointer to the [Measurement](#) (object)

Or

A [Converter Object](#)

value (**double**) - Output Fixed Frequency in Hertz.

Return Type Double

Default Not Applicable

Examples `Print mixer.OutputFixedFrequency 'prints the output fixed frequency value of the mixer.`

C++ Syntax HRESULT get_OutputFixedFrequency(double *pVal)

HRESULT put_OutputFixedFrequency(double newVal)

Interface IMixer3

IConverter

Last Modified:

2-Feb-2009 Added Converter

4-Mar-2008 Added note.

OutputPort Property

Description Switches an input to one of the valid outputs on the specified E5091A. The following are valid input/output combinations. If a combination other than these are sent, an “invalid argument” error will occur.

Input	Valid Outputs
1	A
	T1- If Port 2 already is connected to T1, then Port 2 will be switched to T2.)
2	T1 - If Port 1 already is connected to T1, then Port 1 will be switched to A.
	T2
3	R1+
	R2+
	R3+ If option 007 (7port), R2 is selected.
4	R1-
	R2-
	R3- If option 007 (7port), R2 is selected.

Note: Do not confuse the similar Testset.[OutputPorts Property](#), which sets or gets the port mapping for ALL ports.

VB Syntax *testsets(1).OutputPort (chNum,input) = output*

Variable [\(Type\)](#) - Description

testsets(1) An item from [Testsets](#) (collection)

Learn how to [identify a testset in the collection.](#)

chNum (**Long**) Channel number of the measurement.

input (**Long**) Testset Input port. Choose from 1|2|3|4.

output (Enum as NAE5091OutputPort) Output port to switch to specified Input.
Choose from:

0 or **naE5091PortA** - Port A

1 or **naE5091PortT1** - Port T1

2 or **naE5091PortT2** - Port T2

3 or **naE5091PortR1** - Port R1

4 or **naE5091PortR2** - Port R2

5 or **naE5091PortR3** - Port R3

Return Type Enum

Default Not Applicable

Examples [See Example Program](#)

C++ Syntax HRESULT get_OutputPort(long channelNum, long inputPort,
E5091OutputPort *outPort);
HRESULT put_OutputPort(long channelNum, long inputPort,
E5091OutputPort outPort);

Interface IE5091Testsets

OutputPorts (Cal Set) Property

Description	Returns the port mapping for the Cal Set.
VB Syntax	<i>portMap</i> = <i>calset</i> . OutputPorts
Variable	(Type) - Description
<i>portMap</i>	(String) Variable to store the returned string. The returned values are the physical ports. The POSITION of the returned values corresponds to the logical ports. For example, with an N44xx test set, if the returned string is "PNA 1,TS 2,PNA 2, TS 4" this means: <ul style="list-style-type: none">• PNA 1 is assigned to logical port 1• TS 2 is assigned to logical port 2• PNA 2 is assigned to logical port 3• TS 4 is assigned to logical port 4
<i>calset</i>	A Cal Set object.
Return Type	String
Default	Depends on the test set.
Example	<pre>portMap = calset.OutputPorts</pre>
C++ Syntax	HRESULT get_OutputPorts(BSTR *mapping);
Interface	ICalset5

Last modified:

9/18/06 MQ Added for multiport

OutputPorts Property

Description Sets or gets the port mappings for ALL ports. An “invalid argument” error will occur if you attempt to set an illegal port combination.

Refer to your testset documentation for valid port combinations.

Note: Do not confuse the similar E5091 [OutputPort Property](#), which sets or gets the port mapping for a single port.

VB Syntax *tset*.**OutputPorts**(*chNum*) = *portList*

Variable [\(Type\) - Description](#)

tset A [TestsetControl](#) object.

chNum **(Long)** Channel number of the measurement.

portList **(String)** A comma-separated list of port mappings. Spaces are ignored at the beginning and end of this text, and before or after commas. Space characters in other locations are not ignored.

Return Type String

Default Not Applicable

Examples [See External Testset Program](#)

C++ Syntax HRESULT get_OutputPorts(long channelNum, BSTR *outPorts);
HRESULT put_OutputPorts(long channelNum, BSTR outPorts);

Interface ITestsetControl

OutputRangeMode Property

Description Sets or returns the Output sweep mode.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

Note: There is also a [OutputRangeMode Property](#) on the Converter Object.

VB Syntax *obj*.OutputRangeMode = *value*

Variable [\(Type\)](#) - Description

obj A [Mixer Interface](#) pointer to the [Measurement](#) (object)

value **(Enum as MixerRangeMode)** - Output sweep mode. Choose from:

0 - **mixSwept**

1 - **mixFixed**

Return Type Enum

Default 0 - mixSwept

Examples `mixer.OutputRangeMode = mixSwept`

C++ Syntax HRESULT get_OutputRangeMode(long *pVal)

HRESULT put_OutputRangeMode(long newVal)

Interface IMixer6

Last Modified:

4-Mar-2008 Added note.

OutputRangeMode Property

Description Sets or returns the Output sweep mode.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

Note: There is also a [OutputRangeMode](#) Property on the Mixer Interface.

VB Syntax *obj*.OutputRangeMode = *value*

Variable [\(Type\)](#) - Description

obj A [Converter Object](#)

value (**Enum as NARangeMode**) - Output sweep mode. Choose from:

0 - **naSwept**

1 - **naFixed**

Return Type Enum

Default 0 - **naSwept**

Examples `conv.OutputRangeMode = naSwept`

C++ Syntax HRESULT get_OutputRangeMode(long *pVal)

HRESULT put_OutputRangeMode(long newVal)

Interface IConverter

Last Modified:

18-Feb-2011 New topic

OutputSideband Property

Description Specify whether to select the sum (High) or difference (Low) products.

- When one LO is used: Input (+ or -) LO1 = Output frequency.
- When two LO's are used: IF (+ or -) LO2 = Output frequency. See Also: [IFSideband Property](#)

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

VB Syntax *obj*.**OutputSideband** = *value*

Variable [\(Type\) - Description](#)

obj A [Converter Object](#)

value (**enum as ConverterSideBand**) - Choose from:

0 - or **naLowSide** Minus (-) on the Mixer setup dialog

1 - or **naHighSide** Plus (+) on the Mixer setup dialog

Return Type Enum as ConverterSideBand

Default naLowSide

Examples

```
Print converter.OutputSideband 'prints the value of the
OutputSideband
```

C++ Syntax HRESULT get_OutputSideband(ConverterSideBand *pVal)

HRESULT put_OutputSideband(ConverterSideBand newVal)

Interface IConverter

Last Modified:

26-Mar-2009 New topic

OutputSideband Property

Description Specify whether to select the sum (High) or difference (Low) products.

- When one LO is used: Input + or - LO1 = Output frequency
- When two LOs are used: IF1 + or - LO2 = Output frequency

Use [IFSideband_Property](#) when two LOs are used to determine the IF1 frequency.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

Note: There is also an [OutputSideband_Property](#) on the Converter Object.

VB Syntax *mixer*.**OutputSideband** = *value*

Variable [\(Type\)](#) - Description

mixer A [Mixer](#) (object)

value (**FCASideBand**) -

Choose from:

0 - LOW Minus (-) on the Mixer setup dialog

1 - HIGH Plus (+) on the Mixer setup dialog

Return Type Enum as FCASideBand

Default LOW

Examples `Print mixer.OutputSideband` 'prints the value of the `OutputSideband`

C++ Syntax HRESULT get_OutputSideband(FCASideBand *pVal)

HRESULT put_OutputSideband(FCASideBand newVal)

Interface IMixer

Last Modified:

2-Oct-2008 Clarified 1 vs 2 LOs

4-Mar-2008 Added note.

OutputStartFrequency Property

Description Sets or returns the mixer output start frequency.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

VB Syntax *obj*.OutputStartFrequency = *value*

Variable [\(Type\)](#) - Description

obj A [Mixer Interface](#) pointer to the [Measurement](#) (object)

Or

A [Converter Object](#)

value **(double)** - Output Start Frequency in Hertz.

Return Type Double

Default Not Applicable

Examples

```
Print mixer.OutputStartFrequency 'prints the value of the
OutputStartFrequency
```

C++ Syntax HRESULT get_OutputStartFrequency(double *pVal)

HRESULT put_OutputStartFrequency(double newVal)

Interface IMixer

IConverter

Last Modified:

2-Feb-2009 Added Converter

4-Mar-2008 Added note.

OutputStopFrequency Property

Description Sets or returns the mixer Output Stop frequency.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

VB Syntax *obj*.OutputStopFrequency = *value*

Variable [\(Type\)](#) - Description

obj A [Mixer Interface](#) pointer to the [Measurement](#) (object)

Or

A [Converter Object](#)

value **(double)** - Output stop frequency in Hertz.

Return Type Double

Default Not Applicable

Examples `Print mixer.OutputStopFrequency 'prints the value of the OutputStopFrequency`

C++ Syntax HRESULT get_OutputStopFrequency(double *pVal)

HRESULT put_OutputStopFrequency(double newVal)

Interface IMixer

IConverter

Last Modified:

2-Feb-2009 Added Converter

4-Mar-2008 Added note.

Read-only

Parameter Property

Description	Returns the measurement Parameter. To change the parameter, use <code>meas.ChangeParameter</code>
VB Syntax	<code>measPar = meas.Parameter</code>
Variable	(Type) - Description
<code>measPar</code>	(string) - Variable to store Parameter string
<code>meas</code>	A Measurement (object)
Return Type	String
Default	Not applicable
Examples	<code>measPar = meas.Parameter 'Read</code>
C++ Syntax	<code>HRESULT get_Parameter(BSTR *pVal)</code>
Interface	IMeasurement

Parameter (Embedded LO) Property

Description Returns the name of the parameter of the specified tuning sweep.

VB Syntax *value* = *embedLO*.**Parameter** (*n*)

Variable [\(Type\)](#) - Description

value **(String)** Variable to store the returned data.

embedLO *Diag* An [EmbeddedLODiagnostic](#) (**object**)

n (Long) Tuning sweep number. Use [NumberOfSweeps](#) to find the number of sweeps taken.

Default Not Applicable

Examples `data= embedLO.Parameter 3 'read`

C++ Syntax HRESULT Parameter(long sweep, BSTR * param);

Interface IEmbeddedLODiagnostic

Last Modified:

13-Apr-2007 MX New topic

Read-only

Parent Property

Description Returns a handle to the parent object of the collection object being referred to in the statement. The parent property allows the user to traverse from an object back up the object hierarchy.

VB Syntax *collection*.Parent

Variable [\(Type\)](#) - Description

collection

- [CalFactorSegments collection](#)
- [CalFactorSegmentsPMAR Collection](#)
- [Channels collection](#)
- [E5091Testset Collection](#)
- [ECalModules Collection](#)
- [ExternalDevices Collection](#)
- [ExternalTestsets Collection](#)
- [Measurements collection](#)
- [NaWindows collection](#)
- [PowerLossSegments collection](#)
- [PowerLossSegmentsPMAR_Collection](#)
- [PowerSensors collection](#)
- [Segments collection](#)
- [Traces collection](#)
- [PathConfigurationManager](#)
- [PowerMeterInterfaces Collection](#)

Return Type Object

Default Not Applicable

Examples `parentobj = chans.Parent 'returns a handle to the parent object (Application) of the chans collection. -Read`

C++ Syntax HRESULT get_Parent(IApplication* *pApplication); //IChannels, IChannel, IMeasurements, INAWindows, and IExternalDevices

HRESULT get_Parent(IChannel* *pChannel); //ITraces

HRESULT get_Parent(INAWindow* *pWindow); //ISegments

HRESULT get_Parent(IPowerSensor* *pSensor);
//ICalFactorSegments(PMAR)

HRESULT get_Parent(ISourcePowerCalibrator* *pCalibrator);
//IPowerSensors, IPowerLossSegments, and IPowerMeterInterfaces

Interface All listed above

Last Modified:

31-Jul-2009 Added ExternalDevices (9.0)

5-Mar-2009 Added PowerMeterInterfaces

PassFailLogic Property

Description Sets and reads the logic of the PassFail line on the [HANDLER IO connector](#) (pin 33). [Learn more about Global Pass/Fail.](#)

Note: This line is connected to both the Handler IO and Aux IO in the PNA. Therefore, this command will affect both of these connectors in the same way.

VB Syntax *object*.**PassFailLogic** = *value*

Variable [\(Type\)](#) - **Description**

object **(object)** - An Aux I/O or Handler I/O object

value **(enum as NARearPanelIOLogic)** Choose from:

0 - naPositiveLogic - Causes the PassFail line to have positive logic (high = pass, low = fail).

1 - naNegativeLogic - Causes the PassFail line to have negative logic (high = fail, low = pass).

Return Type Long Integer

Default naPositiveLogic

Examples

```
aux.PassFailLogic = naNegativeLogic 'Write
Text1.Text = aux.PassFailLogic 'Read
```

C++ Syntax

```
HRESULT put_PassFailLogic ( tagNARearPanelIOLogic Mode );
```

```
HRESULT get_PassFailLogic ( tagNARearPanelIOLogic* Mode );
```

Interface

```
IHWAuxIO
```

```
IHWMaterialHandlerIO
```

PassFailMode Property

Description	Sets and reads the mode of the PassFail line on the HANDLER IO connector (pin 33). Learn more about Global Pass/Fail.
VB Syntax	<i>object</i> . PassFailMode = <i>value</i>
Variable	(Type) - Description
<i>object</i>	(object) - An Aux I/O or Handler I/O object
<i>value</i>	(enum as NAPassFailMode) . Choose from: 0 - naDefaultPassNoWaitMode - the line stays in PASS state. When a device fails, then the line goes to fail IMMEDIATELY. 1 - naDefaultPassWaitMode - the line stays in PASS state. When a device fails, then the line goes to fail after the Sweep End line is asserted. 2 - naDefaultFailWaitMode - the line stays in FAIL state. When a device passes, then the line goes to PASS state after the Sweep End line is asserted.
Return Type	Long Integer
Default	0 - naDefaultPassNoWaitMode
Examples	<pre>HWAuxIO.PassFailMode = naDefaultPassNoWaitMode 'Write mode = HWAuxIO.PassFailMode 'Read</pre>
C++ Syntax	<pre>HRESULT put_PassFailMode (tagNAPassFailMode Mode); HRESULT get_PassFailMode (tagNAPassFailMode* Mode);</pre>
Interface	IHWAuxIO IHWMaterialHandlerIO

PassFailPolicy Property

Description	Sets the policy used to determine how global pass/fail is computed. Learn more about Global Pass/Fail.
VB Syntax	<i>object</i> . PassFailPolicy = <i>value</i>
Variable	(Type) - Description
<i>object</i>	(object) - An Aux I/O or Handler I/O object
<i>value</i>	(enum as NAPassFailPolicy) Choose from: 0 - naPolicyAllTests - - Pass/Fail Status returns PASS if all tests on all measurements pass. 1 - naPolicyAllMeas - Pass/Fail Status returns PASS if all measurements have associated tests, and all tests pass. FAIL is returned if even one measurement has no associated limit test. Only those measurements which are not in HOLD mode contribute to the pass/fail result.
Return Type	Long Integer
Default	naPolicyAllTests
Examples	<pre>matHndler.PassFailPolicy = naPolicyAllTests 'Write</pre> <pre>policy = aux.PassFailPolicy 'Read</pre>
C++ Syntax	HRESULT put_PassFailPolicy (tagNAPassFailPolicy Policy); HRESULT get_PassFailPolicy (tagNARearPanelIOLogic* Policy);
Interface	IHWAuxIO4 IHWMaterialHandlerIO2

PassFailScope Property

Description	Sets and reads the Scope of the PassFail line on the HANDLER IO connector (pin 33). Learn more about Global Pass/Fail.
VB Syntax	<i>object</i> . PassFailScope = <i>value</i>
Variable	(Type) - Description
<i>object</i>	(object) - An Aux I/O or Handler IO object
<i>value</i>	(enum NAPassFailScope) Choose from: 0 - naChannelScope - The PassFail line returns to its default state before sweeps on the next channel start. (A channel measurement may require several sweeps.) 1 - naGlobalScope - The PassFail line returns to its default state before the sweeps for the next triggerable channel start. The default state of the PassFail line before a measurement occurs and after a failure occurs is set by the PassFailMode property.
Return Type	enum NAPassFailScope
Default	1 - naGlobalScope
Examples	<pre>HWAuxIO.PassFailScope = naGlobalScope 'Write</pre> <pre>scope = HWAuxIO.PassFailScope 'Read</pre>
C++ Syntax	<pre>HRESULT put_PassFailScope (tagNAPassFailScope Scope);</pre> <pre>HRESULT get_PassFailScope (tagNAPassFailScope* Scope);</pre>
Interface	IHWAuxIO IHWMaterialHandlerIO

PassFailStatus Property

Description Returns the most recent pass/fail status value. Use this command as follows:

1. Set the PNA [trigger scope](#) to GLOBAL
2. Set the PNA [trigger source](#) to MANUAL or EXTERNAL.
3. Configure and enable [Limit Testing](#)
4. Trigger the PNA.
5. Use the `*OPC?` (with [SCPIStringParser object](#)) to determine when the sweep is complete.
6. Use the **PassFailStatus** property to obtain the global pass/fail result.

[Learn more about Global Pass/Fail.](#)

VB Syntax `var = object.PassFailStatus`

Variable [\(Type\)](#) - **Description**

`var` (enum as `NAPassFailStatus`) Variable to store returned status. One of the following will be returned:

0 - naStatusFail - all measurements not in HOLD mode have been swept, and one or more limit tests failed according to the specified [Pass/Fail policy](#).

1 - naStatusPass - all measurements not in HOLD mode have been swept, and all associated limit tests have passed.

2 - naStatusNone - status cannot be determined because measurements are in progress.

object **(object)** - An [Aux I/O](#) or [Handler I/O](#) object

Return Type Long Integer

Default Not Applicable

Examples `status = aux.PassFailStatus 'Read`

C++ Syntax `HRESULT get_PassFailPolicy (tagNAPassFailStatus* status);`

Interface `IHWAuxIO4`

`IHWMaterialHandlerIO2`

Path Property

Description Specifies an interface to use for the power meter / sensor during a source power calibration.

VB Syntax *pwrMtrInterface*.**Path** = *value*

Variable [\(Type\)](#) - Description

pwrMtrInterface **(object)** - A [PowerMeterInterface](#) (object)

value **(enum as NACommunicationPath)** Choose from:

0 - naGPIB - GPIB interface

1 - naUSB - USB interface

2 - naLAN - LAN interface

3 - naAny - Any VISA resource string or a visa alias

Return Type Enum

Default naGPIB

Examples [See example](#)

C++ Syntax HRESULT put_Path(tagNACommunicationPath pNewVal);

HRESULT get_Path(tagNACommunicationPath *pVal);

Interface IPowerMeterInterface

Last Modified:

24-Jan-2012 Added naAny

2-Aug-2007 MX New topic

PathCalMethod Property

Description **Note:** This command replaces [ThruCalMethod](#).

(Read-Write) Specifies the calibration method for each port pair.

Note: Sending this command will overwrite the PNAs SmartCal determinations for the most accurate cal method for your connector settings and Cal Kits. Send this command ONLY if you have a deliberate reason for overwriting the SmartCal logic. You can send the query form of this command to learn the cal method determined by SmartCal.

See [Thru Pairs Sequence](#) to learn how to send this and other Thru commands.

After sending this command, send the query form to be sure that the command was accepted. If not, then the chosen Cal method is not compatible with the specified Thru method. For example, if the specified Thru method is Unknown Thru, an attempt to set Enhanced Response Cal should be rejected.

See an example of a [4-port guided calibration using COM](#).

VB Syntax `guidedCal.PathCalMethod (port1, port2) = "caltype1[,caltype2]"`

Variable [\(Type\) - Description](#)

`guidedCal` [GuidedCalibration](#) (object)

`port1` First port of the pair to be calibrated.

`port2` Second port of the pair to be calibrated.

`"caltype1[,caltype2]"` (String) Cal types for 1st and 2nd ports of the pair, enclosed in a single pair of quotes. NOT case-sensitive.

caltype1 Cal type for the pair if caltype2 is not specified. Otherwise, Cal type for port 1. Choose from:

- "TRL"
- "SOLT"
- "QSOLTN"
- "EnhRespN"

For the last two arguments, replace N with the port to be used as the source port, which MUST be one of the port pair.

[caltype2] Optional argument. Use only when performing an adapter removal cal on the pair. This argument specifies the Cal Type on the second port; *caltype1* then specifies the Cal Type of the first port.

Choose from the same arguments as *caltype1*.

Return Type **String** - Returns comma-separated cal types.

Default The most accurate cal method for the current cal.

Example `guidedCal.PathCalMethod(2,3) = "TRL" 'Write trl for port pair`

```
guidedCal.PathCalMethod(1,4) = "TRL,SOLT" 'Write adapter  
removal cal, consisting of trl on port 1 and solt on port 4
```

```
calmethod = guided.PathCalMethod(1,4) 'Read previous example,  
returns: "TRL,SOLT"
```

C++ Syntax HRESULT get_PathCalMethod(long firstport, long secondport, BSTR *calMethod);

HRESULT put_PathCalMethod(long firstport, long secondport, BSTR calMethod);

Interface IGuidedCalibration3

Last modified:

11-Jan-2013 Added sequence link
6-Apr-2011 Added default note.
April 9, 2007 MX New topic

PathConfigurationElement Property

Description	Allows you to set and return the Path Configuration settings for a Cal All calibration. Learn more about this setting.
VB Syntax	<i>calAll</i> . PathConfigurationElement (<i>element</i>). value = <i>setting</i>
Variable	(Type) - Description
<i>calAll</i>	A CalibrateAllChannels (object)
<i>element</i>	(String) Path configuration element to be set. See a list of configurable RF Path elements.
	Is this also for IF Path configuration elements?
<i>setting</i>	(String) Path configuration element setting.
Return Type	String
Default	Default settings vary with each element.
Examples	<pre>CalAll.PathConfigurationElement("Port1NoiseTuner").value = "Internal"</pre> <pre>'returns the PathConfigurationElement setting value = CalAll.PathConfigurationElement("Port1NoiseTuner").value</pre>
C++ Syntax	HRESULT put_PathConfigurationElement(BSTR name, IPathElement** pElement);
Interface	ICalibrateAllChannels

Last modified:

4-Jan-2012 New topic

PathThruMethod Property

Description **Note:** This command replaces [ThruCalMethod](#).

(Read-Write) Specifies the calibration THRU method for each port pair.

Note: Sending this command will overwrite the PNAs SmartCal determination for the thru method. Send this command **ONLY** if you have a deliberate reason for overwriting the SmartCal logic. You can send the query form of this command to learn the THRU method determined by SmartCal.

See [Thru Pairs Sequence](#) to learn how to send this and other Thru commands.

See an example of a [4-port guided calibration using COM](#).

VB Syntax `guidedCal.PathThruMethod (port1, port2) = "ThruType1[,ThruType2]"`

Variable [\(Type\) - Description](#)

`guidedCal` [GuidedCalibration](#) (object)

`port1` First port of the pair to be calibrated.

`port2` Second port of the pair to be calibrated.

`"ThruType1[,ThruType2]"` (String) Thru methods for 1st and 2nd ports of the pair, enclosed in a single pair of quotes. NOT case-sensitive.

thruType1 Calibration thru method for the pair if thruType2 is not specified. Otherwise, thru method for port 1.

Choose from:

- **“Defined Thru”** Measures a Thru for which there is a stored definition in the Cal Kit.
- **“Zero Thru”** Measures a Zero length Thru, also known as Flush-Thru.
- **“Undefined Thru”** A thru type for which there is NOT a stored definition in the Cal Kit. Also known as Unknown Thru. Valid **ONLY** for SOLT [cal type](#).
- **“Undefined Thru using a Defined Thru”** (ECal modules ONLY) Measures the internal Thru as an Unknown Thru.

ThruType2 (String) Optional argument. Use ONLY when Adapter Removal Cal is specified for the pair using [PathCalMethod](#). When specifying ThruType2, this is the only valid argument: "**Defined Thru, Defined Thru**"

Return Type **String** - Returns comma-separated ThruTypes.

Always returns two parts:

If the second part of the string is empty, adapter removal is NOT being performed.

If the string is "Defined Thru, Defined Thru", adapter removal IS being performed.

Default The most accurate THRU method for the current cal.

Example

```
guidedCal.PathThruMethod(2,3) = "Zero Thru" 'Write for port pair
```

```
guidedCal.PathThruMethod(1,4) = "Defined Thru, Defined Thru" 'Write for adapter removal cal.
```

```
calmethod = guided.PathThruMethod(1,4) 'Read previous example, return: "Defined Thru, Defined Thru"
```

C++ Syntax HRESULT get_PathThruMethod(long firstport, long secondport, BSTR *thruMethod);

HRESULT put_PathThruMethod(long firstport, long secondport, BSTR thruMethod);

Interface IGuidedCalibration3

Last modified:

11-Jan-2013	Added sequence link
6-Apr-2011	Added default note
18-Feb-2011	Replacement command
9-Apr-2007	MX New topic

PeakExcursion Property

Description Sets and reads the peak excursion value for the specified marker. The Excursion value determines what is considered a "peak".

VB Syntax *mark*.PeakExcursion = *value*

Variable [\(Type\)](#) - Description

mark A [Marker](#) (**object**)

value (**single**) - Peak Excursion. Choose any number between **-500** and **500**

Return Type Single

Default 3

Examples

```
mark.PeakExcursion = 1 'Write
```

```
PkExcur = mark.PeakExcursion 'Read
```

C++ Syntax HRESULT get_PeakExcursion(float *pVal)
HRESULT put_PeakExcursion(float newVal)

Interface IMarker

PeakThreshold Property

Description Sets peak threshold for the specified marker. If a peak (using the criteria set with [PeakExcursion](#)) is below this reference value, it will not be considered when searching for peaks.

VB Syntax `mark.PeakThreshold = value`

Variable [\(Type\)](#) - Description

mark A [Marker](#) (object)

value (single) - Peak Threshold. Choose any number between: -500 and 500

Return Type Single

Default -100db

Examples `mark.PeakThreshold = 1 'Write`

`PkThresh = mark.PeakThreshold 'Read`

C++ Syntax HRESULT get_PeakThreshold(float *pVal)
HRESULT put_PeakThreshold(float newVal)

Interface IMarker

PeakToPeak Property

Description	Returns the Peak to Peak value of the measurement. To retrieve all 3 statistics value at the same time, use meas.GetTraceStatistics
VB Syntax	<code>pp = meas.PeakToPeak</code>
Variable	(Type) - Description
<i>pp</i>	(single) - Variable to store peak-to-peak value
<i>meas</i>	A Measurement (object)
Return Type	Single
Default	Not applicable
Examples	<code>pp = meas.PeakToPeak</code> 'Read
C++ Syntax	<code>HRESULT get_PeakToPeak(float* pp)</code>
Interface	IMeasurement

Percent Property

Description Sets group delay aperture using a percent of the channel frequency span.

VB Syntax `gdAperture.Percent = value`

Variable [\(Type\)](#) - Description

gdAperture A [GroupDelayAperture](#) (object)

value **(Double)** Percent of frequency span to use for the aperture setting. Choose between the equivalent of 2 points and 100 percent of the channel frequency span.

Return Type Double

Default Percent range that equates to 11 points.

This can be changed to two points with a [preference setting](#).

Examples

```
gdAperture.Percent = 25 'Write
```

```
aperture = gdAperture.Percent 'Read
```

C++ Syntax HRESULT get_Percent(double Percent *pVal)

HRESULT put_Percent(double Percent newVal)

Interface IGroupDelayAperture

Last Modified:

23-Feb-2010 MX New topic

PerformPowerCalibration Property

Description	Enables Guided Power Cal and sets the source port to be calibrated.
VB Syntax	<i>guidedCal</i> . PerformPowerCalibration (<i>port</i>) = <i>value</i>
Variable	(Type) - Description
<i>guidedCal</i>	GuidedCalibration (object)
<i>port</i>	(Long integer) Source port to be calibrated. ONLY one port may be calibrated with a Guided Power Cal.
<i>value</i>	(Boolean) True Perform a Guided Power Calibration. False Do NOT perform a Guided Power Calibration
Return Type	Boolean
Default	False
Example	<pre>guided.PerformPowerCalibration(1) = True</pre> See example program
C++ Syntax	HRESULT get_PerformPowerCalibration(long port, VARIANT_BOOL* val); HRESULT put_PerformPowerCalibration(long port, VARIANT_BOOL newVal);
Interface	IGuidedCalibration7

Last Modified:

9-Sep-2010 MX New topic

Period Property

Description Sets the pulse-period (1/PRF) for ALL PNA-X internal pulse generators.

The resolution of the period is 16.667nS.

VB Syntax *pulse*.**Period** = *value*

Variable [\(Type\)](#) - Description

pulse A [PulseGenerator](#) (object)

value (**Double**) Pulse period in seconds. Choose a value from about 33ns to about 70 seconds.

Return Type Double

Default 1e-3 sec

Examples

```
pulse.Period = 1ms 'Write
```

```
value = pulse.Period 'Read
```

C++ Syntax HRESULT get_Period(double* period);

HRESULT put_Period(double period);

Interface IPulseGenerator

Last Modified:

1-Jan-2007 MX New topic

PhaseOffset Property

Description Sets the Phase Offset for the active channel.

VB Syntax *meas*.PhaseOffset = *value*

Variable [\(Type\)](#) - Description

meas A Measurement (**object**)

value (**double**) - PhaseOffset in degrees. Choose any number between: **-360** and **+360**

Return Type Double

Default 0

Examples

```
meas.PhaseOffset = 25 'Write
```

```
poffset = meas.PhaseOffset 'Read
```

C++ Syntax HRESULT get_PhaseOffset(double *pVal)
HRESULT put_PhaseOffset(double newVal)

Interface IMeasurement

PhaseAsFixture Property

Description	Sets and reads the state of phase offset as a fixture with True Mode balanced measurements. Learn more about iTMSA phase and power offset.
VB Syntax	<i>balStim.PhaseAsFixture = value</i>
Variable	(Type) - Description
<i>balStim</i>	A BalancedStimulus (object)
<i>value</i>	(Boolean) State of phase offset as a fixture. False Offset is applied but is NOT included as a fixture in the output calculations. True Offset is applied and included as a fixture in the output calculations.
Return Type	Boolean
Default	False
Examples	<pre>balStim.PhaseAsFixture = False 'Write</pre> <pre>var = balStim.PhaseAsFixture 'Read</pre>
C++ Syntax	HRESULT get_PhaseAsFixture (VARIANT_BOOL *bVal) HRESULT put_PhaseAsFixture (VARIANT_BOOL bVal)
Interface	IBalancedStimulus

Last Modified:

27-May-2008 MX New topic

PhaseControlMode Property

Description Write and read the Phase Control mode for the specified port.

VB Syntax *phase*.PhaseControlMode(*srcPort*) = *value*

Variable [\(Type\)](#) - Description

phase A [PhaseControl](#) Object

srcPort (Long Integer) Source port for which to make phase control settings.

Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#)

value (Enum as NPhaseControlMode) Choose from:

0 - naPhaseControlOff Do NOT control the phase of <port>.

1 - naPhaseControlParameter Control the phase of <port>.

2 - naPhaseControlReference (READ-only) - reference port for a 'controlled' (parameter) Port.

3 - naPhaseControlOpenLoop Sets a raw phase value for either swept phase or fixed phase, but no receivers are used to control the phase.

Return Type Enum

Default **0 - naPhaseControlOff**

Examples

```
phase.PhaseControlMode 1 = naPhaseControlParameter ' Write  
value = phase.PhaseControlMode 2 ' Read
```

C++ Syntax HRESULT get_PhaseControlMode(long port, enum NPhaseControlMode* pVal);

HRESULT put_PhaseControlMode(long port, enum NPhaseControlMode newVal);

Interface IPhaseControl

Last Modified:

26-Jun-2012 Changed enum 2

8-Dec-2010 MX New topic

PhaseCorrectionData Property

Description Write and read an array of phase offsets. Each phase offset is summed with each phase point to get new target value. The PNA attempts to set phase for each target value. The number of phase offset values must be the same as the number of data points.

VB Syntax `phase.PhaseCorrectionData(srcPort) = value`

Variable [\(Type\)](#) - Description

phase A [PhaseControl](#) Object

srcPort (Long Integer) Source port for which to make phase control settings.

Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#)

value (Long values) Phase offset data array.

Return Type Long data array

Default Not Applicable

Examples

```
phase.PhaseCorrectionData 1 = 10,11,12 ' Write 3 phase offset values
value = phase.PhaseCorrectionData 2 ' Read
```

C++ Syntax HRESULT get_PhaseCorrectionData(long port, long* pVals);

HRESULT put_PhaseCorrectionData(long port, long newVals);

Interface IPhaseControl

Last Modified:

8-Dec-2010 MX New topic

PhaseCorrectionEnabled Property

Description	Write and read whether to use the phase correction offset array. Use PhaseCorrectionData to write or read the offset data.
VB Syntax	<i>phase</i> .PhaseCorrectionEnabled(<i>srcPort</i>) = <i>value</i>
Variable	(Type) - Description
<i>phase</i>	A PhaseControl Object
<i>srcPort</i>	(Long Integer) Source port for which to make phase control settings.
	Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use chan.getPortNumber to translate the string into a port number. To learn more see Remotely Specifying a Source Port
<i>value</i>	(Boolean) Phase correction array state.
	True – Apply phase correction offset array.
	False – Do NOT apply phase correction offset array.
Return Type	Boolean
Default	False
Examples	<pre>phase.PhaseCorrectionEnabled 1 = True ' Write value = phase.PhaseCorrectionEnabled 2 ' Read</pre>
C++ Syntax	HRESULT get_PhaseCorrectionEnabled(long port, VARIANT_BOOL* pVal); HRESULT put_PhaseCorrectionEnabled(long port, VARIANT_BOOL newVal);
Interface	IPhaseControl

Last Modified:

8-Dec-2010 MX New topic

PhaseIterationNumber Property

Description	Write and read the maximum number of background phase sweeps to perform.
VB Syntax	<i>phase</i> .PhaseIterationNumber(<i>srcPort</i>) = <i>value</i>
Variable	(Type) - Description
<i>phase</i>	A PhaseControl Object
<i>srcPort</i>	(Long Integer) Source port for which to make phase control settings.
	Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use chan.getPortNumber to translate the string into a port number. To learn more see Remotely Specifying a Source Port
<i>value</i>	(Double) Number of background sweep iterations. Choose a value between 1 and 25.
Return Type	Double
Default	5
Examples	<pre>phase.PhaseIterationNumber 1 = 15 ' Write value = phase.PhaseIterationNumber 2 ' Read</pre>
C++ Syntax	HRESULT get_PhaseIterationNumber(long port, double* pVal); HRESULT put_PhaseIterationNumber(long port, double newVal);
Interface	IPhaseControl

Last Modified:

8-Dec-2010 MX New topic

PhaseParameter Property

Description Write and read the ratioed receivers (parameter) and paired port to use for phase control.

VB Syntax *phase.PhaseParameter(srcPort) = value*

Variable [\(Type\)](#) - Description

phase A [PhaseControl](#) Object

srcPort (Long Integer) Source port for which to make phase control settings.

Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#)

value (String) Ratioed parameter. Choose any two PNA physical receivers. Use either standard receiver notation ("R/R3") or [logical receiver notation](#) ("a1/a3"). Separate the two receiver names by a forward slash '/'. For example: "a1/a3".

Return Type String

Default "a1/b1"

Examples

```
phase.PhaseParameter 1 = "a1/a3" ' Write
value = phase.PhaseParameter 2 ' Read
```

C++ Syntax HRESULT get_PhaseParameter(long port, BSTR* pVal);

HRESULT put_PhaseParameter(long port, BSTR newVal);

Interface IPhaseControl

Last Modified:

8-Dec-2010 MX New topic

PhaseParameterModes Property

Description Returns the available phase control modes for the specified port. Use [PhaseControlMode](#) to set the Phase Control mode.

VB Syntax `phase.PhaseParameterModes(srcPort) = value`

Variable [\(Type\)](#) - Description

phase A [PhaseControl](#) Object

srcPort (Long Integer) Source port for which to make phase control settings.

Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#)

value (String) Choose from:

"Off" - Turn phase control OFF

"Openloop" - Sets a raw phase value for either swept phase or fixed phase, but no receivers are used to control the phase.

"Parameter" - Sets and controls the phase of the signal at <port>.

"Reference" - Reference port for a controlled (parameter) port. Use [PhaseReferencePort](#) to set the reference port.

Return Type String

Default "Off"

Example `value = phase.PhaseParameterModes 2 ' Read`

C++ Syntax HRESULT get_PhaseParameterModes(long port, BSTR* pVal);

Interface IPhaseControl

Last Modified:

8-Dec-2010 MX New topic

PhaseReference Property

Description	Sets and returns the Phase Reference ID to be used for the Phase Reference calibration. Use GetConnectedPhaseReferences to read the phase references currently connected to the PNA USB.
VB Syntax	<i>phasRef</i> . PhaseReference = <i>value</i>
Variable	(Type) - Description
<i>phasRef</i>	A PhaseReferenceCalibration Object
<i>value</i>	(String) Phase Reference ID name.
Return Type	String
Default	Not Applicable
Examples	<pre>phase.PhaseReference = "MYPRT0001"</pre> See example program
C++ Syntax	HRESULT get_PhaseReference(BSTR PhaseReference* pVals); HRESULT put_PhaseReference(BSTR PhaseReference newVals);
Interface	IPhaseReferenceCalibration

Last Modified:

3-Apr-2012 MX New topic

PhaseReferencePort Property

Description Sets and returns the reference port for the Phase Control measurement.

VB Syntax *phase*.PhaseReferencePort(*srcPort*) = *value*

Variable [\(Type\)](#) - Description

phase A [PhaseControl](#) Object

srcPort (Long Integer) Source port for which to make phase control settings.

Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#)

value (Long Integer) Reference port number. ONLY specific ports are available to be a reference for each source port. [Learn more](#).

Return Type Long Integer

Default Depends on the controlled port.

Examples

```
phase.PhaseReferencePort 1 = 3 ' Write
value = phase.PhaseReferencePort 2 ' Read
```

C++ Syntax

```
HRESULT get_PhaseReferencePort(long port, long* pVal);
HRESULT put_PhaseReferencePort(long port, long newVal);
```

Interface IPhaseControl

Last Modified:

8-Dec-2010 MX New topic

PhaseSwpAsFixture Property

Description	Sets and reads the state of phase offset as a fixture with True Mode balanced measurements.
VB Syntax	<i>balStim.PhaseAsFixture</i> = <i>value</i>
Variable	(Type) - Description
<i>balStim</i>	A BalancedStimulus (object)
<i>value</i>	(Boolean) State of phase offset as a fixture. False Offset is applied but is NOT included as a fixture in the output calculations. True Offset is applied and included as a fixture in the output calculations.
Return Type	Boolean
Default	False
Examples	<pre>balStim.PhaseAsFixture = False 'Write</pre> <pre>var = balStim.PhaseAsFixture 'Read</pre>
C++ Syntax	HRESULT get_PhaseAsFixture (VARIANT_BOOL *bVal) HRESULT put_PhaseAsFixture (VARIANT_BOOL bVal)
Interface	IBalancedStimulus2

Last Modified:

3-Mar-2009 MX New topic (A.08.50)

PhaseSwpState Property

Description	Sets and reads the state of phase sweep. Sweep type must be set to CWTime.
VB Syntax	<i>balStim.PhaseSwpState</i> = <i>value</i>
Variable	(Type) - Description
<i>balStim</i>	A BalancedStimulus (object)
<i>value</i>	(Boolean) State of phase sweep. False Phase sweep disabled. True Phase sweep enabled.
Return Type	Boolean
Default	False
Examples	<pre>balStim.PhaseSwpState = False 'Write</pre> <pre>var = balStim.PhaseSwpState 'Read</pre>
C++ Syntax	HRESULT get_PhaseSwpState (VARIANT_BOOL *bVal) HRESULT put_PhaseSwpState (VARIANT_BOOL bVal)
Interface	IBalancedStimulus2

Last Modified:

3-Mar-2009 MX New topic (A.08.50)

PhaseTolerance Property

Description Write and read the tolerance value to be used for background phase sweeps.

VB Syntax *phase*.PhaseTolerance(*srcPort*) = *value*

Variable [\(Type\)](#) - Description

phase A [PhaseControl](#) Object

srcPort (Long Integer) Source port for which to make phase control settings.

Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#)

value (Double) Tolerance for background sweeps in degrees. Choose a value between 1 and 5.

Return Type Double

Default 5 degrees

Examples

```
phase.PhaseTolerance 1 = 3 ' Write  
value = phase.PhaseTolerance 2 ' Read
```

C++ Syntax HRESULT get_PhaseTolerance(long port, double* pVal);

HRESULT put_PhaseTolerance(long port, double newVal);

Interface IPhaseControl

Last Modified:

8-Dec-2010 MX New topic

Pin Property

Description Returns the Pin result of a PNOP or PSAT marker search.

PNOP In = Marker 4 X-axis value

PSAT In = Marker 2 X-axis value

VB Syntax *pIn* = *pMark*.**Pin**

Variable [\(Type\)](#) - Description

pIn (**double**) - Variable to store returned value

pMark A [PNOP](#) (object) or [PSaturation](#) (object)

Return Type Double

Default Not applicable

Examples `pIn = pMark.Pin 'Read`

[See example program](#)

C++ Syntax HRESULT get_Pin(double* pNewVal)

Interface IPNOP or IPSaturation

Last Modified:

19-Feb-2010 MX New topic

PinOffset Property

Description Sets and returns the PinOffset value used to calculate various PNOP parameters.

Also set [BackOff Property](#).

A sweep must be executed (single or continuous) and [SearchPowerNormalOperatingPoint Method](#) must be sent before reading marker results.

To turn off the PNOP markers, either turn them off individually or [DeleteAllMarkers](#).

To search a [User Range](#) with the PNOP search, first activate marker 1. The user range used with the PNOP search only applies to marker 1 searching for the linear gain value. The other markers may fall outside the user range.

VB Syntax *pnop.PinOffset = value*

Variable [\(Type\)](#) - Description

pnop A [PNOP](#) (object)

value (**double**) - PinOffset value in dB. Choose any number between: **-500** and **500**

Return Type Double

Default 0 dB

Examples `pinOffs = pnop.PinOffset 'Read`

[See example program](#)

C++ Syntax HRESULT put_PinOffset(double newVal);

HRESULT get_PinOffset(double* pNewVal)

Interface IPNOP

PMaxBackOff Property

Description Sets and returns the backoff value used to calculate various PSAT parameters.

A sweep must be executed (single or continuous) and [SearchPowerSaturation Method](#) must be sent before reading marker results.

To turn off the PSAT markers, either turn them off individually or [DeleteAllMarkers](#).

To search a [User Range](#) with the PSAT search, first activate marker 1. The user range used with the PNOP search only applies to marker 1 searching for the linear gain value. The other markers may fall outside the user range.

VB Syntax *pSat.PMaxBackOff = value*

Variable [\(Type\) - Description](#)

pSat A [PSaturation \(object\)](#)

value **(double)** - Backoff value in dB. Choose any number between:-**500** and **500**

Return Type Double

Default 0 dB

Examples `backoff = pSat.PMaxBackOff 'Read`

[See example program](#)

C++ Syntax HRESULT put_PMaxBackOff(double newVal);

HRESULT get_PMaxBackOff(double* pNewVal)

Interface IPSaturation

Last Modified:

22-Feb-2010 MX New topic

PMaxIn Property

Description Returns the PMaxIn result of a PNOP and PSAT marker search.

PMax In = Marker 3 X-axis value

VB Syntax *pMaxIn* = *pMark*.**PMaxIn**

Variable [\(Type\)](#) - [Description](#)

pMaxIn (**double**) - Variable to store returned value

pMark A [PNOP](#) (object) or [PSaturation](#) (object)

Return Type Double

Default Not applicable

Examples `pMaxIn = pMark.PMaxIn 'Read`

[See example program](#)

C++ Syntax HRESULT get_PMaxIn(double* pNewVal)

Interface IPNOP or IPSAT

Last Modified:

19-Feb-2010 MX New topic

PMaxOut Property

Description Returns the PMaxOut result of the PNOP and PSAT marker search.

PMaxOut = Marker 3 Y-axis value

VB Syntax *pMaxOut* = *pMark*.**PMaxOut**

Variable (Type) - Description

pMaxOut (**double**) - Variable to store returned value

pMark A [PNOP](#) (object) or [PSaturation](#) (object)

Return Type Double

Default Not applicable

Examples `pMaxOut = pMark.PMaxOut 'Read`

[See example program](#)

C++ Syntax HRESULT get_PMaxOut(double* pNewVal)

Interface IPNOP or IPSaturation

Last Modified:

19-Feb-2010 MX New topic

PointAveragingState Property

Description Turns point averaging ON or OFF for all measurements on the channel.

VB Syntax *chan*.**PointAveragingState** = *state*

Variable [\(Type\)](#) - Description

chan A [Channel](#) (object)

state (**Enum**)

False - Turns point averaging OFF

True - Turns point averaging ON

Return Type Boolean

Default False

Examples `chan.PointAveragingState = True 'Write`

`averg = chan.PointAveragingState 'Read`

C++ Syntax HRESULT get_PointAveragingState(BOOL *pVal)
HRESULT put_PointAveragingState(BOOL newVal)

Interface IChannel16

Last Modified:

7-Oct-2008 MX New topic

PointDwell Property

Description Sets and returns the "Dwell Before/After Point" value for an external DC Device which can be configured as either a DC Meter or a DC Source.

VB Syntax `extDC.PointDwell = value`

Variable [\(Type\)](#) - Description

`extDC` An [ExternalDCDevice](#) (**object**)

`value` (**Double**)

For DC Meter, the dwell time (in seconds) before making a data point measurement.

For DC Source, the dwell time (in seconds) after making a data point setting.

Return Type Double

Default 3 milliseconds

Examples

```
extDC.PointDwell = 10e-3 'Write
dwell = extDC.PointDwell 'Read
```

C++ Syntax HRESULT get_PointDwell (double *pValue)

HRESULT put_PointDwell (double newVal)

Interface IExternalDCDevice

Last Modified:

15-Feb-2012 New topic

Points Property

Description Sets group delay aperture using a fixed number of data points.

VB Syntax `gdAperture.Points = value`

Variable [\(Type\)](#) - [Description](#)

gdAperture A [GroupDelayAperture](#) (**object**)

value (**Double**) Number of data points to use for the aperture setting. Choose between two points and the number of points in the channel.

Return Type Double

Default Points range that equates to 11 points.

This can be changed to two points with a [preference setting](#).

Examples `gdAperture.Points = 25 'Write`

`aperture = gdAperture.Points 'Read`

C++ Syntax HRESULT get_Points(double Points *pVal)

HRESULT put_Points(double Points newVal)

Interface IGroupDelayAperture

Last Modified:

23-Feb-2010 MX New topic

PointSweepState Property

Description Turns point sweep ON or OFF for all measurements on the channel. Point sweep measures both the forward and reverse parameters at each frequency point before stepping to the next frequency. The display trace is updated after the forward and reverse parameters are measured at that frequency point.

VB Syntax *chan*.**PointSweepState** = *state*

Variable [\(Type\)](#) - Description

chan A [Channel](#) (object)

state (boolean)

False - Turns point sweep OFF

True - Turns point sweep ON

Return Type Boolean

Default False

Examples `chan.PointSweepState = True 'Write`

`averg = chan.PointSweepState 'Read`

C++ Syntax HRESULT get_PointSweepState(VARIANT_BOOL *pVal)
HRESULT put_PointSweepState(VARIANT_BOOL newVal)

Interface IChannel16

Last Modified:

3-Nov-2008 MX New topic

Port1 Property **Superseded**

Description	This command is replaced by PortDelay property . Sets a Port Extension value for Port 1
VB Syntax	<i>portExt</i> .Port1 = <i>value</i>
Variable	(Type) - Description
<i>portExt</i>	A Port Extension (object)
<i>value</i>	(double) - Port Extension value in seconds. Choose any number between -10 and 10
Return Type	Double
Default	0
Examples	<pre>portExt.Port1 = 10e-6 'Write prt1 = portExt.Port1 'Read</pre>
C++ Syntax	HRESULT get_Port1(double *pVal) HRESULT put_Port1(double newVal)
Interface	IPortExtension

Port2 Property **Superseded**

Description	This command is replaced by PortDelay property . Sets a Port Extension value for Port 2
VB Syntax	<i>portExt</i> .Port2 = <i>value</i>
Variable	(Type) - Description
<i>portExt</i>	A Port Extension (object)
<i>value</i>	(double) - Port Extension value in seconds. Choose any number between -10 and 10
Return Type	Double
Default	0
Examples	<pre>portExt.Port2 = 10e-6 'Write</pre> <pre>prt2 = portExt.Port2 'Read</pre>
C++ Syntax	HRESULT get_Port2(double *pVal) HRESULT put_Port2(double newVal)
Interface	IPortExtension

Port3 Property **Superseded**

Description	This command is replaced by PortDelay property .
	Sets a Port Extension value for Port 3
VB Syntax	<i>portExt</i> .Port3 = <i>value</i>
Variable	(Type) - Description
<i>portExt</i>	A Port Extension (object)
<i>value</i>	(double) - Port Extension value in seconds. Choose any number between -10 and 10
Return Type	Double
Default	0
Examples	<pre>portExt.Port3 = 10e-6 'Write</pre> <pre>prt3 = portExt.Port3 'Read</pre>
C++ Syntax	HRESULT get_Port3(double *pVal) HRESULT put_Port3(double newVal)
Interface	IPortExtension

Port1NoiseTunerSwitchPresetsToExternal Property

Description Sets the default setting for the Noise Tuner switch. [Learn more.](#)

This property returns an error on PNA models with a built-in Noise tuner.

VB Syntax `pref.Port1NoiseTunerSwitchPresetsToExternal = bool`

Variable [\(Type\)](#) - Description

pref A [Preferences](#) (object)

bool (**Boolean**) - Choose from:

False - Sets the default (preset) to INTERNAL

True - Sets the default (preset) to EXTERNAL

Return Type Boolean

Default True

Examples

```
pref.Port1NoiseTunerSwitchPresetsToExternal = False 'Write
```

```
prefer = pref.Port1NoiseTunerSwitchPresetsToExternal 'Read
```

C++ Syntax

```
HRESULT put_Port1NoiseTunerSwitchPresetsToExternal( VARIANT_BOOL bValue)
```

```
HRESULT get_Port1NoiseTunerSwitchPresetsToExternal( VARIANT_BOOL  
*bValue)
```

Interface IPreferences8

Last Modified:

1-Feb-2008 MX New topic

Port2PdeembedCktModel Property

Description Select whether or not to load a 2-port De-embedding circuit model for the specified port number. Circuit model is applied when both "USER" is selected and the filename is specified. To set the filename, use [strPort2Pdeembed_S2PFile Property](#)

Note: This command affects ALL measurements on the channel.

VB Syntax `fixture.Port2PdeembedCktModel(port) = value`

Variable [\(Type\) - Description](#)

fixture A [Fixturing](#) (object)

port (**Integer**) Port number to receive circuit model.

value (**Enum as NAFixturing2PdeembedCkt**)

0 **naFix2PD_USER** load a 2-port De-embedding circuit model

1 **naFix2PD_NONE** no model

Return Type Long Integer

Default naFix2PD_NONE

Examples

```
fixture.Port2PdeembedCktModel(2) = naFix2PD_USER 'Write
value = fixture.Port2PdeembedCktModel(1) 'Read
```

C++ Syntax HRESULT get_Port2PdeembedCktModel(short port tagNAFixturing2PdeembedCkt *pVal)
 HRESULT put_Port2PdeembedCktModel(short port tagNAFixturing2PdeembedCkt newVal)

Interface IFixturing

Port2PdeembedState Property

Description	Turns de-embedding ON or OFF for all ports on the channel.
VB Syntax	<i>fixture</i> .Port2PdeembedState = <i>value</i>
Variable	(Type) - Description
<i>fixture</i>	A Fixturing (object)
<i>value</i>	(Boolean)
	False - Turns De-embedding OFF
	True - Turns De-embedding ON
Return Type	Boolean
Default	False
Examples	<pre>fixture.Port2PdeembedState = False 'Write</pre> <pre>value = fixture.Port2PdeembedState 'Read</pre>
C++ Syntax	HRESULT get_Port2PdeembedState(VARIANT_BOOL *pVal) HRESULT put_Port2PdeembedState(VARIANT_BOOL newVal)
Interface	IFixturing

PortArbzImag Property

Description Sets and returns the Imaginary portion of the impedance value for the specified single-ended port. Use [PortArbzReal](#) to set the real value. Or use [PortArbzZ0](#) to set both values together.

VB Syntax *fixture.PortArbzImag(portNum) = value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

portNum (**Integer**) Single-ended port number to receive impedance value.

value (**Double**) Real Impedance value. Choose a value between -1E18 and 1E18

Return Type Double

Default 0

Examples `fixture.PortArbzImag(2) = 75 'Write`

`value = fixture.PortArbzImag(1) 'Read`

C++ Syntax HRESULT get_PortArbzImag(short portNum, double *pVal)
HRESULT put_PortArbzImag(short portNum, double newVal)

Interface IFixturing3

PortArbzReal Property

Description Sets and returns the Real portion of the impedance value for the specified single-ended port. Use [PortArbzImag](#) to set the imaginary value. Or use [PortArbzZO](#) to set both values together.

VB Syntax *fixture.PortArbzReal(portNum) = value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

portNum (**Integer**) Single-ended port number to receive impedance value.

value (**Double**) Real Impedance value. Choose a value between 0 to 1E7

Return Type Double

Default 50

Examples `fixture.PortArbzReal(2) = 75 'Write`
`value = fixture.PortArbzReal(1) 'Read`

C++ Syntax HRESULT get_PortArbzReal(short portNum, double *pVal)
HRESULT put_PortArbzReal(short portNum, double newVal)

Interface IFixturing3

PortArbzState Property

Description Turns Port Impedance ON or OFF for all ports on the channel.

VB Syntax *fixture*.PortArbzState = *value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

value (**Boolean**)

False - Turns Port Impedance OFF

True - Turns Port Impedance ON

Return Type Boolean

Default False

Examples `fixture.PortArbzState = False 'Write`

`value = fixture.PortArbzState 'Read`

C++ Syntax HRESULT get_PortArbzState(VARIANT_BOOL *pVal)
HRESULT put_PortArbzState(VARIANT_BOOL newVal)

Interface IFixturing

PortArbzZ0 Property

Description Sets and returns the Real portion of the impedance value for the specified single-ended port. The imaginary portion is automatically set to 0.0.

To set both values separately, use [PortArbzReal](#) and [PortArbzImag](#).

VB Syntax `fixture.PortArbzZ0(portNum) = value`

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

portNum (**Integer**) Single-ended port number to receive impedance value.

value (**Double**) Impedance value. Choose a value between 0 to 1E7

Return Type Double

Default 50

Examples `fixture.PortArbzZ0(2) = 75 'Write`

`value = fixture.PortArbzZ0(1) 'Read`

C++ Syntax HRESULT get_PortArbzZ0(short portNum, double *pVal)
HRESULT put_PortArbzZ0(short portNum, double newVal)

Interface IFixturing3

PortCatalog Property

Description	Returns a comma-separated list of the Output port selections that are available for a given logical input port. Read the number of input ports for the test set using NumberOfPorts Property .
VB Syntax	<i>value</i> = <i>tset</i> . PortCatalog (<i>logPort</i>)
Variable	(Type) - Description
<i>value</i>	(String) Variable to store the returned information.
<i>tset</i>	A TestsetControl object.
<i>logPort</i>	(Long) Logical Input port number for which to return valid output ports.
Return Type	String
Default	Not Applicable
Examples	<pre>value = testset1.PortCatalog 2</pre> See External Testset Program
C++ Syntax	HRESULT get_PortCatalog(long inputPort, BSTR *outPort);
Interface	ITestsetControl

Last Modified:

18-Jan-2007 Fixed argument

PortCLogic Property

Description Sets and reads the logic mode of Port C on the Handler IO connector.

VB Syntax *AuxIO.PortCLogic = value*

Variable [\(Type\)](#) - Description

AuxIO **(object)** - A Hardware Aux I/O object

value **(Enum as NaRearPanelIOLogic)** - Choose from:

0 - naPositiveLogic - The associated data line goes **HIGH** when writing a 1 to a PortC bit.

1 - naNegativeLogic - The associated data line goes **LOW** when writing a 1 to a PortC bit.

When Port C is in Output/Write mode, a change in logic causes the output lines to change state immediately. For example, Low levels change to High levels.

When Port C is in Input/Read mode, a change in logic will not cause the lines to change, but data read from Port C will reflect the change in logic.

Return Type Enum

Default 1 - naNegativeLogic

Examples

```
auxIO.PortCLogic = value 'Write
value = auxIo.PortCLogic 'Read
```

C++ Syntax

```
HRESULT put_PortCLogic ( tagNARearPanelIOLogic Mode );
```

```
HRESULT get_PortCLogic ( tagNARearPanelIOLogic* Mode );
```

Interface

```
IHWAuxIO
```

PortCMode Property

Description	Sets and reads whether Port C is setup for writing or reading data on the Handler IO connector.
VB Syntax	<i>AuxIO.PortCMode = value</i>
Variable	(Type) - Description
<i>AuxIO</i>	(object) - A Hardware Aux I/O object
<i>value</i>	(enum as NaPortMode) - Choose from: 0 - naInput - set the port for reading 1 - naOutput - set the port for writing
Return Type	Enum as NaPortMode
Default	1 - naInput
Examples	<pre>auxIo.get_PortCMode = naInput 'Write</pre> <pre>value = auxIo.get_PortCMode 'Read</pre>
C++ Syntax	<pre>HRESULT get_PortCMode(tagNAPortMode* pMode);</pre> <pre>HRESULT put_PortCMode(tagNAPortMode pMode);</pre>
Interface	IHWAuxIO

PortCoupleToSystemMedia Property

Description Sets and returns the state of coupling with the system Media type.

VB Syntax *fixture*.PortCoupleToSystemMedia (*port*)= *value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) Object

port (Integer) Port Number that will receive the coupling change.

value (Boolean) Coupling state. Choose from:

- **True** - Media type is coupled with the system setting.
- **False** - Media type is NOT coupled with the system setting.

Return Type Boolean

Default True

Examples

```
fixture.PortCoupleToSystemMedia(2)= True
```

```
bool = fixture.PortCoupleToSystemMedia(1)
```

C++ Syntax

```
HRESULT get_PortCoupleToSystemMedia(short portNum, VARIANT_BOOL *pVal);
```

```
HRESULT put_PortCoupleToSystemMedia(short portNum, VARIANT_BOOL  
newVal);
```

Interface IFixturing4

Last Modified:

6-Feb-2009 MX New topic

PortCoupleToSystemVelocity Property

Description Sets and returns the state of coupling with the system Velocity Factor.

VB Syntax *fixture*.PortCoupleToSystemVelocity (*port*)= *value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) Object

port (Integer) Port Number that will receive the coupling change.

value (Boolean) Coupling state. Choose from:

- **True** - Velocity Factor is coupled with the system setting.
- **False** - Velocity Factor is NOT coupled with the system setting.

Return Type Boolean

Default True

Examples `fixture.PortCoupleToSystemVelocity(2)= True`

`bool = fixture.PortCoupleToSystemVelocity(1)`

C++ Syntax HRESULT get_PortCoupleToSystemVelocity(short portNum, VARIANT_BOOL *pVal);

HRESULT put_PortCoupleToSystemVelocity(short portNum, VARIANT_BOOL newVal);

Interface IFixturing4

Last Modified:

6-Feb-2009 MX New topic

PortDelay Property

Description Sets and returns the Port Extensions Delay value for the specified port number.

Note: This command affects ALL measurements on the channel.

This command replaces [Port 1](#) [Port 2](#) [Port_3](#) Properties.

VB Syntax `fixture.PortDelay(port) = value`

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

port **(Integer)** Port number to receive delay value.

value **(Double)** Delay value in seconds. Choose a value between -1E18 and 1E18.

Return Type Double

Default 0

Examples `fixture.PortDelay(2) = .002 'Write`

`value = fixture.PortDelay(1) 'Read`

C++ Syntax HRESULT get_PortDelay(short port double *pVal)
HRESULT put_PortDelay(short port double newVal)

Interface IFixturing

PortDescription Property

Description For each port of the ECal module that is going to be characterized, sets and reads the description of the adapters, cable, or fixture to be included in the user characterization. This description is stored with the characterization in the ECal module.

Set this description before sending [Initialize](#) or the default (empty string) will be used.

VB Syntax `userChar.PortDescription (port)= value`

Variable [\(Type\)](#) - Description

userChar An [ECalUserCharacterizer](#) Object

port (Enum) ECal port for which description is to be set. Choose from:

1 or **naECalPort_A**

2 or **naECalPort_B**

3 or **naECalPort_C**

4 or **naECalPort_D**

value (String) Descriptive text, limited to 24 characters maximum.

Return Type String

Default "" (Empty String)

Examples

```
userChar.PortDescription (naECalPort_C)= "3.5 mm adapter, SN  
00001"
```

C++ Syntax HRESULT get_PortDescription(NAECalPort port, *BSTR description);

HRESULT put_PortDescription(NAECalPort port, BSTR description);

Interface IECalUserCharacterizer

Last Modified:

2-Nov-2008 New topic (8.33)

PortDistance Property

Description Sets and returns the port extension delay in physical length (distance).

VB Syntax *fixture*.**PortDistance** (*port*)= *value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) Object

port (Integer) Port Number that will receive the delay setting.

value (Double) Physical length in distance. First specify units with [PortDistanceUnit](#).

Return Type Double

Default 0

Examples `fixture.PortDistance(2)= .01 'Write`

`value = fixture.PortDistance(2) 'Read`

C++ Syntax HRESULT get_PortDistance(short portNum, double *pVal);

HRESULT put_PortDistance(short portNum, double newVal);

Interface IFixturing4

Last Modified:

6-Feb-2009 MX New topic

PortDistanceUnit Property

Description Sets and returns the units for specifying port extension delay in physical length (distance).

VB Syntax *fixture*.PortDistanceUnit = *value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) Object

value (enum naDistanceUnit) Choose from:

0 – naDistanceUnit_Meter

1 – naDistanceUnit_Feet

2 – naDistanceUnit_Inch

Return Type Enum

Default naDistanceUnit_Meter

Examples `fixture.PortDistanceUnit(2) = naDistanceUnit_Meter 'Write`

`value = fixture.PortDistanceUnit(2) 'Read`

C++ Syntax HRESULT get_PortDistanceUnit(short portNum, tagNADistanceUnit*
*pVal);

HRESULT put_PortDistanceUnit(short portNum, tagNADistanceUnit
newVal);

Interface IFixturing4

Last Modified:

6-Feb-2009 MX New topic

PortExtState Property

Description Turns Port Extension ON or OFF for all ports on the channel.

VB Syntax *fixture*.PortExtState = *value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

value (**Boolean**)

False - Turns Port Extensions OFF

True - Turns Port Extensions ON

Return Type Boolean

Default False

Examples `fixture.PortExtState = 0 'Write`

`value = fixture.PortExtState 'Read`

C++ Syntax HRESULT get_PortExtState(VARIANT_BOOL *pVal)
HRESULT put_PortExtState(VARIANT_BOOL newVal)

Interface IFixturing

PortExtUse1 Property

Description Sets and returns the Use1 ON/OFF state for the use of the [PortLoss1](#) and [PortFreq1](#) values for the specified port number.

Note: This command affects ALL measurements on the channel.

VB Syntax *fixture.PortExtUse1(port) = value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

port (**Integer**) Port number to receive Use1 ON / OFF state.

value (**Boolean**)

False - Turns Use1 OFF

True - Turns Use1 ON

Return Type Boolean

Default False

Examples `fixture.PortExtUse1(2) = False 'Write`

`value = fixture.PortExtUse1(1) 'Read`

C++ Syntax HRESULT get_PortExtUse1(short port VARIANT_BOOL *pVal)
HRESULT put_PortExtUse1(short port VARIANT_BOOL newVal)

Interface IFixturing

PortExtUse2 Property

Description Sets and returns the Use2 ON/OFF state for the use of the [PortLoss2](#) and [PortFreq2](#) values for the specified port number.

Note: This command affects ALL measurements on the channel.

VB Syntax *fixture.PortExtUse2(port) = value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

port **(Integer)** Port number to receive Use2 ON / OFF state.

value **(Boolean)**

False - Turns Use1 OFF

True - Turns Use1 ON

Return Type Boolean

Default False

Examples `fixture.PortExtUse2(2) = False 'Write`

`value = fixture.PortExtUse2(1) 'Read`

C++ Syntax HRESULT get_PortExtUse2(short port VARIANT_BOOL *pVal)
HRESULT put_PortExtUse2(short port VARIANT_BOOL newVal)

Interface IFixturing

PortFreq1 Property

Description Sets and returns Frequency1 value for the specified port number.

Note: This command affects ALL measurements on the channel.

VB Syntax `fixture.PortFreq1(port) = value`

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

port (**Integer**) Port number to receive extrapolated loss.

value (Double) Frequency1 value. Choose a frequency within the frequency span of the PNA.

Return Type Double

Default 1 GHz

Examples

```
fixture.PortFreq1(2) = naFix2PD_USER 'Write
```

```
value = fixture.PortFreq1(1) 'Read
```

C++ Syntax HRESULT get_PortFreq1(short port double *pVal)
HRESULT put_PortFreq1(short port double newVal)

Interface IFixturing

PortFreq2 Property

Description Sets and returns Frequency2 value for the specified port number.

Note: This command affects ALL measurements on the channel.

VB Syntax `fixture.PortFreq2(port) = value`

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

port (**Integer**) Port number to receive extrapolated loss.

value (**Double**) Frequency2 value. Choose a frequency within the frequency span of the PNA.

Return Type Double

Default 1 GHz

Examples `fixture.PortFreq2(2) = 10E9 'Write`

`value = fixture.PortFreq2(1) 'Read`

C++ Syntax HRESULT get_PortFreq2(short port double *pVal)
HRESULT put_PortFreq2(short port double newVal)

Interface IFixturing

PortLabel Property

Description Sets and returns the label on the calibration kit Port for the calibration wizard.

VB Syntax `calKit.PortLabel (portNum) = value`

Variable [\(Type\)](#) - Description

calKit A CalKit (**object**)

portNum (**long integer**) - number of the port to be labeled. Choose either **1** or **2**

value (**string**) - Label that is visible in the calibration wizard.

Return Type String

Default Depends on the Cal Kit.

Examples `calKit.PortLabel = "MyCalKit" 'Write`

`kitLabel = calKit.PortLabel 'Read`

C++ Syntax HRESULT get_PortLabel(long port, BSTR *pVal)
HRESULT put_PortLabel(long port, BSTR newVal)

Interface ICalKit

PortLogic Property

Description	Sets and returns the logic mode of data ports A-H on the HandlerIO connector. Port C of the Handler IO is connected internally to the Port C of the Aux IO connector. Therefore, it will have the same logic mode.
VB Syntax	<i>handler</i> . PortLogic = <i>value</i>
Variable	(Type) - Description
<i>handler</i>	(object) - A HandlerI/O object
<i>value</i>	(enum as NaRearPanelIOLogic) - Choose from: 0 - naPositiveLogic - When a value of one is written, the associated line goes High 1 - naNegativeLogic - When a value of one is written, the associated line goes Low For ports that are in output (write) mode, a change in logic causes the output lines to change state immediately. For example, Low levels change immediately to High levels. For ports that are in input (read) mode (C,D,E only), a change in logic will be reflected when data is read from that port. For example, if a line read 0, the next read after a logic change will read 1.
Return Type	Long Integer
Default	1 - naNegativeLogic
Examples	<pre>handler.PortLogic = value 'Write value = handler.PortLogic 'Read</pre>
C++ Syntax	<pre>HRESULT put_PortLogic(tagNARearPanelIOLogic Mode); HRESULT get_PortLogic(tagNARearPanelIOLogic* Mode);</pre>
Interface	IHWMaterialHandlerIO

PortLoss1 Property

Description Sets and returns the Loss1 value for the specified port number.

Note: This command affects ALL measurements on the channel.

VB Syntax *fixture*.PortLoss1(*port*) = *value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

port **(Integer)** Port number to receive Loss value

value **(Double)** Loss1 value in dB. Choose a value between -90 and 90.

Return Type Double

Default 0

Examples

```
fixture.PortLoss1(2) = .002 'Write
```

```
value = fixture.PortLoss1(1) 'Read
```

C++ Syntax HRESULT get_PortLoss1(short port double *pVal)
HRESULT put_PortLoss1(short port double newVal)

Interface IFixturing

PortLoss2 Property

Description Sets and returns the Loss2 value for the specified port number.

Note: This command affects ALL measurements on the channel.

VB Syntax *fixture*.PortLoss2(*port*) = *value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

port **(Integer)** Port number to receive Loss value

value **(Double)** Loss2 value in dB. Choose a value between -90 and 90.

Return Type Double

Default 0

Examples `fixture.PortLoss2(2) = .002 'Write`

`value = fixture.PortLoss2(1) 'Read`

C++ Syntax HRESULT get_PortLoss2(short port double *pVal)
HRESULT put_PortLoss2(short port double newVal)

Interface IFixturing

PortLossDC Property

Description Sets and returns the Loss value at DC for the specified port number.

[Learn about Loss compensation values.](#)

Note: This command affects ALL measurements on the channel.

VB Syntax *fixture.PortLossDC(port) = value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

port **(Integer)** Port number to receive Loss value at DC.

value **(Double)** Loss value in ohms. Choose a value between -90 and 90

Return Type Double

Default 0

Examples

```
fixture.PortLossDC(2) = .002 'Write
```

```
value = fixture.PortLossDC(1) 'Read
```

C++ Syntax HRESULT get_PortLossDC(short port double *pVal)
HRESULT put_PortLossDC(short port double newVal)

Interface IFixturing

PortMatching_C Property

Description Sets and returns the Capacitance value for the specified port number.

Note: This command affects ALL measurements on the channel.

VB Syntax `fixture.PortMatching_C(port) = value`

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

port **(Integer)** Port number to receive capacitance value

value **(Double)** Capacitance value in farads. Choose a value between -1E18 to 1E18.

Return Type Double

Default 0

Examples

```
fixture.PortMatching_C(2) = .00002 'Write
```

```
value = fixture.PortMatching_C(1) 'Read
```

C++ Syntax HRESULT get_PortMatching_C(short port double *pVal)
HRESULT put_PortMatching_C(short port double newVal)

Interface IFixturing

PortMatching_G Property

Description Sets and returns the Conductance value for the specified port number.

Note: This command affects ALL measurements on the channel.

VB Syntax `fixture.PortMatching_G(port) = value`

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

port **(Integer)** Port number to receive conductance value.

value **(Double)** Conductance value in siemens. Choose a value between -1E18 and 1E18.

Return Type Double

Default 0

Examples `fixture.PortMatching_G = .002 'Write`

`value = fixture.PortMatching_G 'Read`

C++ Syntax HRESULT get_PortMatching_G(short port double *pVal)
HRESULT put_PortMatching_G(short port double newVal)

Interface IFixturing

PortMatching_L Property

Description Sets and returns the Inductance value for the specified port number.

Note: This command affects ALL measurements on the channel.

VB Syntax `fixture.PortMatching_L(port) = value`

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

port **(Integer)** Port number to receive inductance value

value **(Double)** Inductance value in henries. Choose a value between -1E18 and 1E18.

Return Type Double

Default 0

Examples

```
fixture.PortMatching_L = .002 'Write  
value = fixture.PortMatching_L 'Read
```

C++ Syntax HRESULT get_PortMatching_L(short port double *pVal)
HRESULT put_PortMatching_L(short port double newVal)

Interface IFixturing

PortMatching_R Property

Description Sets and returns the Resistance value for the specified port number.

Note: This command affects ALL measurements on the channel.

VB Syntax `fixture.PortMatching_R(port) = value`

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

port **(Integer)** Port number to receive resistance value.

value **(Double)** Resistance value in ohms. Choose a value between -1E18 and 1E18.

Return Type Double

Default 0

Examples

```
fixture.PortMatching_R = .1 'Write
value = fixture.PortMatching_R 'Read
```

C++ Syntax HRESULT get_PortMatching_R(short port double *pVal)
HRESULT put_PortMatching_R(short port double newVal)

Interface IFixturing

PortMatchingCktModel Property

Description Sets and returns the Port Matching circuit model for the specified port number.

Note: This command affects ALL measurements on the channel.

VB Syntax `fixture.PortMatchingCktModel(port) = value`

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

port (**Integer**) Port number to receive circuit model.

value (**Enum as NAFixturingPortMatchCkt**) Circuit model. Choose from

0 **naFixPMC_SLPC** Series L - Parallel C

1 **naFixPMC_PCSL** Parallel C - Series L

2 **naFixPMC_PLSC** Parallel L - Series C

3 **naFixPMC_SCPL** Series C - Parallel L

4 **naFixPMC_PLPC** Parallel L - Parallel C

5 **naFixPMC_USER** Load S2P file - also set filename to load with [strPortMatch_S2PFile Property](#)

6 **naFixPMC_NONE** No circuit model

Return Type Long Integer

Default **naFixPMC_NONE**

Examples `fixture.PortMatchingCktModel(2) = naFixPMC_PLSC 'Write`

`value = fixture.PortMatchingCktModel(1) 'Read`

C++ Syntax HRESULT get_PortMatchingCktModel(short port tagNAFixturingPortMatchCkt *pVal)
 HRESULT put_PortMatchingCktModel(short port tagNAFixturingPortMatchCkt newVal)

Interface IFixturing

PortMatchingState Property

Description Sets and returns the Port Matching State on the channel.

VB Syntax *fixture*.PortMatchingState = *value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

value (boolean)

True - Turns Port Matching ON

False - Turns Port Matching OFF

Return Type Boolean

Default False

Examples `fixture.PortMatchingState = True 'Write`

`value = fixture.PortMatchingState 'Read`

C++ Syntax HRESULT get_PortMatchingState(VARIANT_BOOL *pVal)
HRESULT put_PortMatchingState(VARIANT_BOOL newVal)

Interface IFixturing

PortMedium Property

Description Sets and returns the media type of the added fixture or transmission line.

VB Syntax *fixture*.**PortMedium** (*port*)= *value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) Object

port (Integer) Port Number for which media type is being set.

value (enum NACalStandardMedium) - Medium of the transmission line of the standard. Choose from:

0 - naCoax - Coaxial Cable

1 - naWaveGuide - Waveguide

Return Type Enum

Default **0 - naCoax**

Examples `fixture.PortMedium(2)= naCoax 'Write`

`value = fixture.PortMedium(2) 'Read`

C++ Syntax HRESULT get_PortMedium(short portNum, tagNACalStandardMedium* pVal);

HRESULT put_PortMedium(short portNum, tagNACalStandardMedium newVal);

Interface IFixturing4

Last Modified:

6-Feb-2009 MX New topic

PortMode Property

Description	Sets and returns whether Port C or Port D is used for writing or reading data on the Handler IO connector. The Handler IO Port C is connected internally to the Port C of the Aux IO connector. Therefore, the Aux IO connector will have the same input/output mode.
VB Syntax	<i>handler</i> . PortMode (<i>port</i>) = <i>value</i>
Variable	(Type) - Description
<i>handler</i>	(object) - A Handler I/O object
<i>port</i>	(enum as NAMatHandlerPort) Port to be changed. Choose from: 2 - naPortC 3- naPortD
<i>value</i>	(enum as NaPortMode) - Choose from: 0 - naInput - set the port for reading 1 - naOutput - set the port for writing
Return Type	Long Integer
Default	1 - naInput
Examples	<pre>handler.PortMode(naPortC) = naInput 'Write value = handler.PortMode(naPortD) 'Read</pre>
C++ Syntax	<pre>HRESULT put_PortMode (tagNAMatHandlerPort Port, tagNAPortMode Mode); HRESULT get_PortMode (tagNAMatHandlerPort Port, tagNAPortMode* Mode);</pre>
Interface	IHWMaterialHandlerIO

PortsNeedingDeltaMatch Property

Description Returns the port numbers for which delta match correction is required. 0 (zero) is returned if the Cal does NOT require Delta Match correction for one of the following reasons:

- The Cal does NOT involve Unknown Thru or TRL. You specify this using [ThruCalMethod Property](#).
- The Cal DOES involve Unknown Thru or TRL, but the delta match data can be calculated by the Unknown Thru or TRL Cal. [Learn how this is possible](#). However, you can force the Cal to use the Delta Match data from a Cal Set.

VB Syntax *value* = *guided*.PortsNeedingDeltaMatch

Variable [\(Type\)](#) - Description

value (Variant) Variable to store the returned list of port numbers.

guided [GuidedCalibration](#) (object)

Return Type Variant

Default Not Applicable

Examples

```
Dim ports As Variant
ports = guided.PortsNeedingDeltaMatch
```

C++ Syntax HRESULT get_PortsNeedingDeltaMatch (VARIANT* portList);

Interface IGuidedCalibration2

PortVelocityFactor Property

Description Sets and returns the Port Extensions Velocity Factor value for the specified port number.

Note: This command affects ALL measurements on the channel.

VB Syntax `fixture.PortVelocityFactor(port) = value`

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

port **(Integer)** Port number to receive velocity factor value.

value **(Double)** Velocity Factor value. Choose a number between: 0 and 10
(.66 polyethylene dielectric; .7 PTFE dielectric)

Return Type Double

Default 1

Examples

```
fixture.PortVelocityFactor(2) = .6 'Write
```

```
value = fixture.PortVelocityFactor(1) 'Read
```

C++ Syntax HRESULT get_PortVelocityFactor(short port double *pVal)
HRESULT put_PortVelocityFactor(short port double newVal)

Interface IFixturing4

Last Modified:

12-Feb-2009 MX New topic

PortWGCutoffFreqProperty

Description Sets and returns the cutoff (minimum) frequency of the added waveguide fixture or transmission line.

VB Syntax *fixture.PortWGCutoffFreq(port)= value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) Object

port (Integer) Port Number for which media type is being set.

value (Double) Cutoff frequency in Hz.

This value is ignored when [PortMedium Property](#) is set to **COAX** for the same port.

Return Type Double

Default System Media Cutoff Frequency

Examples `fixture.PortWGCutoffFreq(2)= 1e9 'Write`

`value = fixture.PortWGCutoffFreq(2) 'Read`

C++ Syntax HRESULT get_PortWGCutoffFreq(short portNum, double *pVal);

HRESULT put_PortWGCutoffFreq(short portNum, double newVal);

Interface IFixturing4

Last Modified:

6-Feb-2009 MX New topic

POut Property

Description Returns the POut result of the PNOP or PSAT marker search.

PSAT Out = Marker 2 Y-axis value

PNOPOut = Marker 4 Y-axis value

VB Syntax *pOut* = *pMark*.**POut**

Variable [\(Type\)](#) - [Description](#)

pOut (**double**) - Variable to store returned value

pMark A [PNOP](#) (Object) or [PSaturation](#) (Object)

Return Type Double

Default Not applicable

Examples `pOut = pMark.POut 'Read`

[See example program](#)

C++ Syntax HRESULT get_POut(double* pNewVal)

Interface IPNOP or IPSaturation

Last Modified:

19-Feb-2010 MX New topic

PowerSlope Property

Description Sets or returns the Power Slope value. Power Slope function increases or decreases the output power over frequency. Units are db/GHz. For example: PowerSlope = 2 will increase the power 2db/1GHZ.

VB Syntax *object*.PowerSlope = *value*

Variable [\(Type\)](#) - Description

object [Channel](#) (**object**)

or

[CalSet](#) (**object**) - Read-only property

value (**double**) - Power Slope. Choose any number between -2 and 2.
No slope = 0

Return Type Double

Default 0

Examples

```
chan.PowerSlope = 2 'Write
```

```
pwrslp = chan.PowerSlope 'Read
```

C++ Syntax HRESULT get_PowerSlope(double *pVal)
HRESULT put_PowerSlope(double newVal)

Interface IChannel
|CalSet3

PowerAcquisitionDevice Property

Description Returns the power sensor channel (A or B) that is currently selected for use at a specific frequency.

If [UsePowerSensorFrequencyLimits](#) is set to False, this property will return the sensor channel last used for a source power calibration. This setting corresponds to the **Use this sensor only** checkbox in the [Power Sensor Settings](#) dialog.

When performing an SMC calibration, use [SetPowerAcquisitionDevice Method](#) to set the power sensor channel.

VB Syntax *sensor* = *pwrCal*.PowerAcquisitionDevice(*dFreq*)

Variable [\(Type\)](#) - Description

sensor (**enum NPowerAcquisitionDevice**) The currently selected sensor channel for the specified frequency. Choose from:

0 – naPowerSensor_A

1 – naPowerSensor_B

pwrCal A SourcePowerCalibrator (object)

dFreq (double) Frequency (Hz) for the power reading of interest.

Return Type enum NPowerAcquisitionDevice

Default Not Applicable

Examples `selectedSensor = pwrCal.PowerAcquisitionDevice(1.E9) 'Write`

C++ Syntax HRESULT get_PowerAcquisitionDevice(double dFreq, tagNPowerAcquisitionDevice* enumAcqDevice);

Interface ISourcePowerCalibrator2

PowerAsFixture Property

Description Sets and reads the state of power offset as a fixture with True Mode balanced measurements. [Learn more about iTMSA power and power offset.](#)

VB Syntax *balStim.PowerAsFixture = value*

Variable [\(Type\)](#) - Description

balStim A [BalancedStimulus](#) (object)

value (**Boolean**) State of power offset as a fixture.

False Offset is applied but is NOT included as a fixture in the output calculations.

True Offset is applied and included as a fixture in the output calculations.

Return Type Boolean

Default False

Examples

```
balStim.PowerAsFixture = False 'Write
```

```
var = balStim.PowerAsFixture 'Read
```

C++ Syntax HRESULT get_PowerAsFixture (VARIANT_BOOL *bVal)

HRESULT put_PowerAsFixture (VARIANT_BOOL bVal)

Interface IBalancedStimulus

Last Modified:

27-May-2008 MX New topic

PowerCalibrationLevel Property

Description Set and read the power level at which to perform the source cal during an Enhanced Power Cal or during the power cal portion of some Noise Figure Cals.

VB Syntax *object*.PowerCalibrationLevel (*port*) = *value*

Variable [\(Type\)](#) - Description

object A [GuidedCalibration](#) (Object)

port **(Long)** PNA Port number to connect the power sensor.

value **(Double)** - Power level in dB. Choose a value from +30 to (-30).

Return Type Double

Default 0

Examples

```
cal.PowerCalibrationLevel(1) = -5 'Write
```

```
pLevel = nfx.PowerCalibrationLevel(1) 'Read
```

[See enhanced power cal example](#)

C++ Syntax HRESULT get_PowerCalibrationLevel(long port, double* pVal)
HRESULT put_PowerCalibrationLevel(long port, double* pVal)

Interface IGuidedCalibration6

Last Modified:

4-Oct-2010 Added enhanced power cal

12-Oct-2009 MX New topic

PowerCalibrationPowerLevel Property

Description Set and read the power level at which to perform the source cal during an Guided Power Cal or during an NFX Cal.

When used with [Guided Power Cal](#), first enable a power cal using [PerformPowerCalibration Property](#).

VB Syntax *object.PowerCalibrationPowerLevel* (*port*) = *value*

Variable [\(Type\)](#) - Description

object A [GuidedCalibration](#) (Object)

port (**Long**) PNA Port number to connect the power sensor.

value (**Double**) - Power level in dB. Choose a value from +30 to (-30).

Return Type Double

Default 0

Examples `cal.PowerCalibrationPowerLevel(1) = -5 'Write`

`pLevel = nfx.PowerCalibrationPowerLevel(1) 'Read`

C++ Syntax HRESULT get_PowerCalibrationPowerLevel(long port, double* pVal)

HRESULT put_PowerCalibrationPowerLevel(long port, double* pVal)

Interface IGuidedCalibration6

Last Modified:

12-Oct-2009 MX New topic

PowerLevel Property

Description Set and read the power level at which to perform the Source Power Cal portion of a Gain Compression Calibration or a SweptIMD Cal.

VB Syntax *object*.**PowerLevel** = *value*

Variable [\(Type\)](#) - Description

object A [GainCompressionCal](#) (**object**)

A [SweptIMDCal](#) (**object**)

value (**Double**) - Power level in dB. Choose a value from +30 to (-30). [Learn about choosing a power level.](#)

Return Type Double

Default 0

Examples

```
gca.PowerLevel = -5 'Write
pLevel = imd.PowerLevel 'Read
```

C++ Syntax HRESULT get_PowerLevel(double* pVal)
HRESULT put_PowerLevel(double newVal)

Interface IGainCompressionCal
ISweptIMDCal

Last Modified:

9-Sep-2008 Added IMD
11-Apr-2008 MX New topic

PowerLevel (Cal All) Property

Description Sets and returns the power level at which a Cal All calibration is to be performed.

VB Syntax `calAll.PowerLevel (port) = value`

Variable [\(Type\)](#) - Description

`calAll` A [CalibrateAllChannels](#) (object)

`port` (Long) Source port number.

`value` (Double) Power level at which the calibration is to be performed.

Return Type Double

Default ??

Examples

```
calAll.PowerLevel = 0 'Power Level of cal
```

```
value = calAll.PowerLevel 'returns the power level of the cal
```

C++ Syntax HRESULT get_PowerLevel (long port, double val);

HRESULT put_PowerLevel long port, double* newVal);

Interface ICalibrateAllChannels

Last modified:

4-Jan-2012 New topic

PowerMax Property

Description Sets and returns the maximum power level for Safe Mode.

VB Syntax *RxLevel.PowerMax(srcPort) = value*

Variable [\(Type\)](#) - Description

RxLevel A [ReceiverLeveling](#) Object

srcPort (Long Integer) Source port for which to set the max power for Receiver Leveling.

Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#)

value (Double) Maximum power level in dB.

Return Type (Double)

Default + 30

Examples

```
rxLevel.PowerMax (1) = 25 ' Write
value = rxLevel.PowerMax 2 ' Read
```

C++ Syntax

```
HRESULT get_PowerMax(long port, double* pVal);
```

```
HRESULT put_PowerMax(long port, double newVal);
```

Interface IReceiverLevelingConfiguration

Last Modified:

13-Feb-2009 MX New topic

PowerMaxIn

PowerMaxOut

PowerMeterChannel Property

Description	Identifies which channel of the power meter the power sensor is connected to.
VB Syntax	<i>chan</i> = <i>powerSensor</i> . PowerMeterChannel
Variable	(Type) - Description
<i>chan</i>	(enum NAPowerAcquisitionDevice) – Power meter channel identifier for sensor. Choose from: 0 – naPowerSensor_A 1 – naPowerSensor_B
<i>pwrSensor</i>	A PowerSensor (Object) or A PowerSensorAsReceiver (Object)
Return Type	NAPowerAcquisitionDevice
Default	Not Applicable
Examples	<pre>meterChannel = pwrSensor.PowerMeterChannel</pre>
C++ Syntax	HRESULT PowerMeterChannel(tagNAPowerAcquisitionDevice *pSensor);
Interface	IPowerSensor IPowerSensorAsReceiver

Last Modified:

25-Aug-2009 Added PMAR

PowerMeterGPIBAddress Property **Superseded**

Description This command is replaced with [PowerMeterInterface Object](#).

Specifies the GPIB address of the power meter that will be referenced by the SourcePowerCalibrator object.

When performing a source power cal, the PNA will search VISA interfaces that are configured in the Agilent IO Libraries on the PNA.

VB Syntax *powerCalibrator*.**PowerMeterGPIBAddress** = *value*

Variable [\(Type\)](#) - Description

powerCalibrator **(object)** - A SourcePowerCalibrator (object)

value **(long integer)** – Power meter GPIB address. Choose any number between 0 and 30.

Return Type Long integer

Default 13

Examples

```
Set powerCalibrator = pna.SourcePowerCalibrator
powerCalibrator.PowerMeterGPIBAddress = 13 'Write
pwrMtrAddress = powerCalibrator.PowerMeterGPIBAddress 'Read
```

C++ Syntax HRESULT put_PowerMeterGPIBAddress(long newVal);

HRESULT get_PowerMeterGPIBAddress(long *pVal);

Interface ISourcePowerCalibrator

Last Modified:

9-Jul-2007 Superseded

PowerMin Property

Description Sets and returns the minimum power level for Safe Mode.

VB Syntax *RxLevel*.**PowerMin**(*srcPort*) = *value*

Variable [\(Type\)](#) - Description

RxLevel A [ReceiverLeveling](#) Object

srcPort (Long Integer) Source port for which to set the min power for Receiver Leveling.

Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#)

value (Double) Minimum power level in dB.

Return Type (Double)

Default -95 dB

Examples

```
rxLevel.PowerMin (1) = -50 ' Write  
value = rxLevel.PowerMin 2 ' Read
```

C++ Syntax

```
HRESULT get_PowerMin(long port, double* pVal);
```

```
HRESULT put_PowerMin(long port, double newVal);
```

Interface IReceiverLevelingConfiguration

Last Modified:

13-Feb-2009 MX New topic

PowerOffset Property

Description Sets and returns the power level offset value.

VB Syntax *RxLevel*.**PowerOffset**(*srcPort*) = *value*

Variable [\(Type\)](#) - Description

RxLevel A [ReceiverLeveling](#) Object

srcPort (Long Integer) Source port for which to set the offset power for Receiver Leveling.

Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#)

value (Double) Power level Offset in dB.

Return Type (Double)

Default 0 dB

Examples

```
rxLevel.PowerOffset (1) = 10 ' Write  
value = rxLevel.PowerOffset 2 ' Read
```

C++ Syntax

```
HRESULT get_PowerOffset(long port, double* pVal);
```

```
HRESULT put_PowerOffset(long port, double newVal);
```

Interface IReceiverLevelingConfiguration

Last Modified:

13-Feb-2009 MX New topic

PowerOffset (Cal All) Property

Description	Sets and returns the power offset value for a Cal All calibration. Power Offset provides a method of compensating port power for added attenuation or amplification in the source path. The result is that power at the specified port reflects the added components.
VB Syntax	<i>calAll</i> . PowerOffset (<i>port</i>) = <i>value</i>
Variable	(Type) - Description
<i>calAll</i>	A CalibrateAllChannels (object)
<i>port</i>	(Long) Source port number.
<i>value</i>	(Double) Power offset value in dB for a Cal All calibration. <ul style="list-style-type: none">• For amplification, use positive offset.• For attenuation, use negative offset.
Return Type	Double
Default	0 (zero dB)
Examples	<pre>calAll.PowerOffset = 0 'Power Level of cal</pre> <pre>value = calAll.PowerOffset 'returns the power offset of the cal</pre>
C++ Syntax	HRESULT get_PowerOffset (long port, double val); HRESULT put_PowerOffset long port, double* newVal);
Interface	CalibrateAllChannels

Last modified:

4-Jan-2012 New topic

PowerOnDuringRetraceMode Property

Description For single-band frequency or segment sweeps ONLY, specify whether to turn RF power ON or OFF during a retrace.

This setting remains until changed using this command, or until the hard drive is changed or reformatted.

VB Syntax *pref.PowerOnDuringRetraceMode = value*

Variable [\(Type\)](#) - Description

pref A [Preferences \(object\)](#)

value **(Enum)** - Choose from:

0 - **naRetracePowerMode_Auto** Power is left ON during retrace of single-band frequency or segment sweeps ONLY.

1 - **naRetracePowerMode_OFF** Power is turned OFF during retrace of single-band frequency or segment sweeps ONLY.

Return Type Enum

Default 0 - naRetracePowerMode_Auto

Examples

```
pref.PowerOnDuringRetraceMode = naPowerSweepRetraceMode_Start 'Write
psMode = pref.naPowerOnDuringRetraceMode_Start 'Read
```

C++ Syntax HRESULT get_PowerOnDuringRetraceMode (tagNARetracePowerMode* preference);
 HRESULT put_PowerOnDuringRetraceMode (tagNARetracePowerMode val)

Interface IPreferences4

Last modified:

Nov. 16, 2006 New command

PowerSensorCalKitType Property

Description Set and read the cal kit to be used for calibrating at the reference plane when the power sensor connector is different from the DUT port.

When used with [Guided Power Cal](#), first enable a power cal using [PerformPowerCalibration Property](#).

VB Syntax *object.PowerSensorCalKitType* (*n*) = *value*

Variable [\(Type\)](#) - [Description](#)

object A [GuidedCalibration](#) (**object**)

n (**Long**) PNA port number for which cal kit is specified.

value (**String**) - Cal Kit. Use [CompatibleCalKits Property](#) to return a list of valid cal kits.

Return Type String

Default Not Applicable

Examples `gguided.PowerSensorCalKitType(1) = "85052B" 'Write`

`cType =guided.PowerSensorCalKitType(1) 'Read`

C++ Syntax HRESULT get_PowerSensorCalKitType(long port, BSTR* Val)

HRESULT put_PowerSensorCalKitType(long port, BSTR newVal)

Interface IGuidedCalibration6

Last Modified:

30-Nov-2010 Fixed example

12-Oct-2009 MX New topic

PowerSensorCalkitType Property

Description Set and read the cal kit to be used for calibrating at the adapter when the power sensor connector is different from the DUT. Use [PowerSensorConnectorType](#) to specify the connector type of the adapter.

VB Syntax *object.PowerSensorCalkitType = value*

Variable [\(Type\)](#) - Description

object A [GainCompressionCal](#) (object)

A [SweptIMDCal](#) (object)

A [GuidedCalibrationPowerSensor](#) (object)

value **(String)** - Cal Kit. Use [CompatibleCalKits Property](#) to return a list of valid cal kits.

Return Type String

Default Not Applicable

Examples

```
gca.PowerSensorCalkitType = "85052B" 'Write
ctype = imd.PowerSensorCalkitType 'Read
```

C++ Syntax HRESULT get_PowerSensorCalkitType(BSTR* Val)
 HRESULT put_PowerSensorCalkitType(BSTR newVal)

Interface IGainCompressionCal2
 ISweptIMD
 IGuidedCalibrationPowerSensor

Last Modified:

8-Feb-2011	Added Guided
9-Sep-2008	Added Swept IMD
11-Apr-2008	MX New topic

PowerSensorConnectorType Property

Description Set and read the power sensor connector type which is used to perform the Source Power Cal. Use [PowerSensorCalKitType](#) to specify the Cal Kit to use for the cal.

VB Syntax *object*.PowerSensorConnectorType = *value*

Variable [\(Type\)](#) - Description

object. A [GainCompressionCal](#) (object)

A [SweptIMDCal](#) (object)

A [GuidedCalibrationPowerSensor](#) (object)

value **(String)** - Power sensor connector type. Use [ValidConnectorType Property](#) to return a list of valid connector types.

Select "Ignored" to NOT compensate for the adapter.

Return Type String

Default Not applicable

Examples

```
gca.PowerSensorConnectorType = "APC3.5 male" 'Write
ctype = imd.PowerSensorConnectorType 'Read
```

C++ Syntax HRESULT get_PowerSensorConnectorType(BSTR* Val)

HRESULT put_PowerSensorConnectorType(BSTR newVal)

Interface IGainCompressionCal2

ISweptIMDCal

IGuidedPowerCalPowerSensor

Last Modified:

8-Feb-2011	Added Guided
9-Sep-2008	Added Swept IMD
11-Apr-2008	MX New topic

PowerSensorConnectorType Property

Description Set and read the power sensor connector type which is used to perform a Power Cal during an S-parameter calibration or during an NFX Cal.

When used with [Guided Power Cal](#), first enable a power cal using [PerformPowerCalibration Property](#).

VB Syntax *guided.PowerSensorConnectorType* (*n*) = *value*

Variable [\(Type\)](#) - [Description](#)

guided A [GuidedCalibration](#) (object)

n (**Long**) PNA port number to connect power sensor to.

value (**String**) - Power sensor connector type. Use [ValidConnectorType Property](#) to return a list of valid connector types.

Set to "**Ignored**" to NOT compensate for the adapter.

Return Type String

Default Not applicable.

Examples

```
guided.PowerSensorConnectorType(1) = "APC3.5 male" 'Write
```

```
ctype = guided.PowerSensorConnectorType(1) 'Read
```

C++ Syntax HRESULT get_PowerSensorConnectorType (long port, BSTR* val);

HRESULT put_PowerSensorConnectorType (long port, BSTR newVal);

Interface IGuidedCalibration6

Last Modified:

30-Nov-2010 Added index to example

12-Oct-2009 MX New topic

PowerSlopeState Property

Description Turns point slope ON or OFF for all measurements on the channel. Use [PowerSlope](#) to set the slope value.

VB Syntax `chan.PowerSlopeState = state`

Variable [\(Type\)](#) - Description

chan A [Channel](#) (object)

state (boolean)

False - Turns power slope OFF

True - Turns point slope ON

Return Type Boolean

Default False

Examples `chan.PowerSlopeState = True 'Write`

`state = chan.PowerSlopeState 'Read`

C++ Syntax HRESULT get_PowerSlopeState(VARIANT_BOOL *pVal)

HRESULT put_PowerSlopeState(VARIANT_BOOL newVal)

Interface IChannel18

Last Modified:

1-Apr-2011 New topic

PowerStep Property

Description Sets and returns the safe mode power step value.

VB Syntax *RxLevel.PowerStep(srcPort) = value*

Variable [\(Type\)](#) - Description

RxLevel A [ReceiverLeveling](#) Object

srcPort (Long Integer) Source port for which to set the power step for Receiver Leveling.

Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#)

value (Double) Power step in dB.

Return Type (Double)

Default 1 dB

Examples

```
rxLevel.PowerStep (1) = 2 ' Write  
value = rxLevel.PowerStep 2 ' Read
```

C++ Syntax

```
HRESULT get_PowerStep(long port, double* pVal);
```

```
HRESULT put_PowerStep(long port, double newVal);
```

Interface IReceiverLevelingConfiguration

Last Modified:

13-Feb-2009 MX New topic

PowerSweepRetracePowerMode Property

Description At the end of a power sweep, while waiting to trigger the next sweep, maintain source power at either the start power level or at the stop power level.

VB Syntax *pref.PowerSweepRetracePowerMode = value*

Variable [\(Type\)](#) - Description

pref A [Preferences](#) (object)

value (Enum) - Choose from:

0 - **naPowerSweepRetraceMode_Start** - maintain source at start power level.

1 - **naPowerSweepRetraceMode_Stop** - maintain source at stop power level.

Return Type Enum

Default 0 - naPowerSweepRetraceMode_Start

Examples

```
pref.PowerSweepRetracePowerMode = naPowerSweepRetraceMode_Start 'Write
psMode = pref.PowerSweepRetracePowerMode 'Read
```

C++ Syntax HRESULT get_PowerSweepRetracePowerMode (tagNAPowerSweepRetraceMode* preference);
 HRESULT put_PowerSweepRetracePowerMode (tagNAPowerSweepRetraceMode val)

Interface IPreferences3

Last modified:

30-Aug-2010 Fixed Read example

Oct. 25, 2006 New command

PreciseTuningTolerance Property

Description Sets and returns the tuning tolerance for precise tuning.

VB Syntax *obj.PreciseTuningTolerance = value*

Variable [\(Type\)](#) - Description

obj An [EmbeddedLO](#) (**object**) or

A [ConverterEmbeddedLO](#) (**object**)

value (**Double**) Tuning tolerance in Hz.

Return Type (**Double**)

Default 1 Hz

Examples

```
embedLO.PreciseTuningTolerance = .5 'write
```

```
value = embedLO.PreciseTuningTolerance 'read
```

C++ Syntax HRESULT get_PreciseTuningTolerance(double* tolerance);

HRESULT put_PreciseTuningTolerance(double tolerance);

Interface IEmbeddedLO

Last Modified:

12-Aug-2009 Added ConvEmbedLO object

18-Apr-2007 MX New topic

PreferInternalTriggerOnChannelSingle Property

Description Set and read the preference for the [chan.Single](#) trigger behavior. This setting persists until changed.

These preferences are important when performing a Guided calibration, as the PNA uses the **chan.Single** trigger command to measure standards.

- set `PreferInternalTriggerOnChannelSingle = False` to use an External trigger sweep to measure a cal standard.
- set `PreferInternalTriggerOnChannelSingle = True` to use an External sweep for the measurement, but rely on the PNA to send Internal trigger signals for calibrating.

To set this preference for an **Unguided** Calibration, use [PreferInternalTriggerOnUnguidedCal Property](#)

The **chan.Single** trigger command NEVER respects the Trigger Source = Manual setting. It always switches to Internal for one trigger, then back to Manual, regardless of this preference command.

VB Syntax `pref.PreferInternalTriggerOnChannelSingle = bool`

Variable [\(Type\)](#) - Description

pref A [Preferences](#) (object)

bool (**Boolean**) - Choose from:

0 - False - the Single trigger property does respect the **Trigger Source = External** setting. For example, if [Trigger source = External](#), the single trigger method will wait for the External trigger signal and then allow only one sweep.

1 - True - the Single trigger command does NOT respect the **Trigger Source = External** setting. For example, when the Single method is sent, the PNA immediately switches to Internal sweep, responds to one trigger signal, then switches back to External.

Return Type Boolean

Default 0 - False

Examples

```
pref.PreferInternalTriggerOnChannelSingle = False 'Write
prefer = pref.PreferInternalTriggerOnChannelSingle 'Read
```

C++ Syntax HRESULT put_PreferInternalTriggerOnChannelSingle(VARIANT_BOOL bprefSingle)

HRESULT get_PreferInternalTriggerOnChannelSingle(VARIANT_BOOL
*bprefSingle)

Interface IPreferences2

PreferInternalTriggerOnUnguidedCal Property

Description Set and read the preference for the trigger behavior when performing an Unguided calibration.

VB Syntax `pref.PreferInternalTriggerOnUnguidedCal = bool`

Variable [\(Type\)](#) - Description

pref A [Preferences](#) (object)

bool (**Boolean**) - Choose from:

0 - False - The trigger behavior during an Unguided calibration DOES respect the setting of the [Trigger source](#) command. For example, during an Unguided Cal, when Trigger source = External, the PNA will wait for the External trigger signal and then allow only one sweep.

1 - True -The trigger behavior during an Unguided calibration does NOT respect the **Trigger Source = External** setting. For example, during an Unguided Cal, when Trigger source = External, the PNA immediately switches to Internal sweep, measures the standard with one trigger signal, then switches back to External trigger.

Note: When Trigger Source = Manual during a calibration, the PNA ALWAYS switches to Internal for one trigger to measure a standard, then back to Manual, regardless of this preference command.

Return Type Boolean

Default 0 - False

Examples

```
pref.PreferInternalTriggerOnUnguidedCal = False 'Write
```

```
prefer = pref.PreferInternalTriggerOnUnguidedCal 'Read
```

C++ Syntax HRESULT put_PreferInternalTriggerOnUnguidedCal(VARIANT_BOOL
bprefUnguided)

HRESULT get_PreferInternalTriggerOnUnguidedCal(VARIANT_BOOL
*bprefUnguided)

Interface IPreferences2

PresetPowerState Property

Description Set and return the Preset Power ON/OFF state.

VB Syntax *pref.PresetPowerState* = *value*

Variable [\(Type\)](#) - Description

pref A [Preferences](#) (object)

value (Enum as NAPowerStates) - Choose from:

naPowerON (0) - Instrument Preset always turns RF power ON.

naPowerAUTO (1):

- When the current power setting is OFF, leave power OFF at Preset.
- When the current power setting is ON, turn power ON at Preset.

Return Type enum

Default naPowerON (0)

Examples

```
pref.PresetPowerState = naPowerON 'Write
```

```
pwrState = pref.PresetPowerState 'Read
```

C++ Syntax HRESULT get_PresetPowerState(tagNAPowerStates* pVal);
HRESULT put_PresetPowerState(tagNAPowerStates pVal);

Interface IPreferences11

Last Modified:

9-Mar-2010 MX New topic

PropertyNames Property

Description	Returns the unique, settable properties for the current cal all session. See a list of valid properties and values for each measurement class.
VB Syntax	<i>props</i> = <i>calAll</i> . PropertyNames
Variable	(Type) - Description
<i>props</i>	(Variant Array) Variable to store the returned property names that can be set.
<i>calAll</i>	A CalibrateAllChannels (object)
Return Type	Variant Array
Default	Not Applicable
Examples	<pre>props = calAll.PropertyNames</pre>
C++ Syntax	HRESULT get_PropertyNames (VARIANT*, propName);
Interface	CalibrateAllChannels

Last modified:

10-Feb-2014 Added link to props/values

PropertyValue Property

Description Sets and returns the property value for a specific property name.

[See a list of valid properties and values for each measurement class.](#)

Use [PropertyValues](#) to query a list of valid values.

VB Syntax `calAll.PropertyValue (propName) = value`

Variable [\(Type\)](#) - **Description**

`calAll` A [CalibrateAllChannels](#) (**object**)

`propName` (String) Property name for which value is to be set or returned.

`value` (String) Property value.

Return Type String

Default Not Applicable

Examples `calAll.PropertyValue ("Noise Cal Method") = "Scalar"`

C++ Syntax `HRESULT get_PropertyValue (BSTR propName, BSTR* propValue);`

`HRESULT put_PropertyValue (BSTR propName, BSTR propValue);`

Interface CalibrateAllChannels

Last modified:

10-Feb-2014 Added link to prop/values

19-Sep-2012 Minor Mods

4-Jan-2012 New topic

PropertyValues Property

Description Returns the valid property values for a specific property name.

[See a list of valid properties and values for each measurement class.](#)

VB Syntax *props* = *calAll*.**PropertyValues** (*propName*)

Variable [\(Type\)](#) - Description

props (Variant Array) Variable to store the returned property values that can be set.

calAll A [CalibrateAllChannels](#) (**object**)

propName (String) Property name for which valid values are to be returned.

Return Type Variant Array

Default Not Applicable

Examples `props = calAll.PropertyValues ("Noise Cal Method")`

C++ Syntax HRESULT get_PropertyValues (BSTR propName, VARIANT* propValues);

Interface CalibrateAllChannels

Last modified:

10-Feb-2014 Added link to props/values

19-Sep-2012 Minor mods

4-Jan-2012 New topic

PulseGeneratorID Property

Description	Returns the ID of the specified External Pulse Generator name. Use this ID number when setting properties on the PulseGenerator Object . Use PulseGeneratorNames to read the names of the internal and configured external pulse generators.
VB Syntax	<i>value</i> = <i>chan</i> . PulseGeneratorID (<i>name</i>)
Variable	(Type) - Description
<i>value</i>	(Long Integer) - Variable to store the returned ID.
<i>chan</i>	A Channel (object)
<i>name</i>	Name of the pulse generator. Use PulseGeneratorNames to read the names of configured pulse generators.
Return Type	Long Integer
Default	Not Applicable
Example	<pre>id = chan.PulseGeneratorID "My81110"</pre>
C++ Syntax	HRESULT get_PulseGeneratorID(long* count, BSTR name)
Interface	IChannel23

Last modified:

16-Feb-2012 New topic

PulseGeneratorNames Property

Description Returns the string names of internal and configured external pulse generators.

VB Syntax *names* = *chan*.**PulseGeneratorNames**

Variable [\(Type\)](#) - Description

names Variant array to store the returned string names.

chan A [Channel](#) (**object**)

Return Type Variant Array of string names

Default Not applicable

Example `names = extPulseGen.PulseGeneratorNames 'Read`

C++ Syntax HRESULT get_PulseGeneratorNames (VARIANT *pNames)

Interface IChannel

Last Modified:

15-Feb-2012 New topic

PulseMeasMode Property

Description Sets the pulse measurement state for the channel.

VB Syntax *pulseMeas.PulseMeasMode* = *value*

Variable [\(Type\)](#) - Description

pulseMeas A [PulseMeasurementControl](#) (object)

value (Enum as NAPulseMeasurementMode) Choose from:

(0) **naPulseMeasurementOff** - Turn OFF pulse measurements.

(1) **naPulseStandardMeasurement** - Turn ON standard pulse measurements.

(2) **naPulseProfileMeasurement** - Turn ON pulse profile measurements.

Return Type Enum

Default (0) **naPulseMeasurementOff**

Examples

```
pulse.PulseMeasMode = naPulseProfileMeasurement 'Write  
value = pulse.PulseMeasMode 'Read
```

C++ Syntax HRESULT get_PulseMeasMode(tagNAPulseMeasurementMode* pVal);
HRESULT put_PulseMeasMode(tagNAPulseMeasurementMode newVal);

Interface IPulseMeasurementControl

Last Modified:

11-Mar-2010 New topic

RangeCount Property

Description	Returns the number of ranges that are available in the PNA. To see the range names, query the Name property of each range in the FOM collection.
VB Syntax	<i>value</i> = FOM.RangeCount
Variable	(Type) - Description
<i>object</i>	An FOM (collection object)
<i>value</i>	(long) - Variable to store the returned number of ranges.
Return Type	Long Integer
Default	Not Applicable
Examples	<pre>NumRanges = fom.RangeCount 'Read</pre>
C++ Syntax	HRESULT get_RangeCount(long *count)
Interface	IFOM

Last Modified:

8-Mar-2007 Added link to Name property

rangeNumber Property

Description	Returns the index number of the range within the FOM collection .
VB Syntax	<i>value</i> = <i>FOMRange</i> .rangeNumber
Variable	(Type) - Description
<i>value</i>	(Long) - Variable to store the returned range number.
<i>object</i>	An FOMRange (object)
Return Type	Long
Default	Not Applicable
Examples	<pre>num = fomRange.rangeNumber 'Read</pre>
C++ Syntax	HRESULT get_rangeNumber(long *pVal)
Interface	IFOMRange

RatioedPowerCorrectionData Property

Description Write and read an array of ratioed power offsets. This allows the setting of arbitrary impedance, which is used for active load applications. The number of offset values must be the same as the number of data points. Use [RatioedPowerCorrectionEnabled](#) to use the offset values.

VB Syntax `phase.RatioedPowerCorrectionData(srcPort) = value`

Variable [\(Type\)](#) - Description

phase A [PhaseControl](#) Object

srcPort (Long Integer) Source port for which to make phase control settings.

Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#)

value (Long values) Ratioed amplitude offset data array.

Return Type Long data array

Default Not Applicable

Examples

```
phase.RatioedPowerCorrectionData 1 = .1,.2,.3 ' Write 3 power
offset values
value = phase.RatioedPowerCorrectionData 2 ' Read
```

C++ Syntax HRESULT get_RatioedPowerCorrectionData(long port, long* pVals);

HRESULT put_RatioedPowerCorrectionData(long port, long newVals);

Interface IPhaseControl

Last Modified:

8-Dec-2010 MX New topic

RatioedPowerCorrectionEnabled Property

Description Write and read whether to use the ratioed power offset array. Use [RatioedPowerCorrectionData Property](#) to write or read the offset data.

VB Syntax `phase.RatioedPowerCorrectionEnabled(srcPort) = value`

Variable [\(Type\)](#) - Description

phase A [PhaseControl](#) Object

srcPort (Long Integer) Source port for which to make phase control settings.

Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#)

value (Boolean) Ratioed power offset array state.

True – Apply offset array.

False – Do NOT apply offset array.

Return Type Boolean

Default False

Examples

```
phase.RatioedPowerCorrectionEnabled 1 = True ' Write  
value = phase.RatioedPowerCorrectionEnabled 2 ' Read
```

C++ Syntax

```
HRESULT get_RatioedPowerCorrectionEnabled(long port,  
VARIANT_BOOL* pVal);
```

```
HRESULT put_RatioedPowerCorrectionEnabled(long port,  
VARIANT_BOOL newVal);
```

Interface IPhaseControl

Last Modified:

8-Dec-2010 MX New topic

RcvCharMethod Property

Description Set and read the method used to characterize the noise receivers.

VB Syntax *noise*.**RcvCharMethod** = *value*

Variable [\(Type\)](#) - Description

noise A [NoiseCal](#) (object)

value (**string**) Receiver characterization method. NOT case-sensitive. Choose from:

- **"NoiseSource"** - Use a noise source. This selection is NOT allowed when a standard PNA receiver is used as the noise receiver. ([noise.NoiseReceiver](#) = naStandardReceiver).
- **"PowerMeter"** - Use a power meter.

NOTE: The Power Meter selection is **NOT** allowed when the Noise Bandwidth is **8 MHz** or **24 MHz**.

Return Type String

Default "NoiseSource"

Examples

```
noise.RcvCharMethod = "PowerMeter" 'Write  
receiverMethod = noise.RcvCharMethod 'Read
```

C++ Syntax HRESULT get_RcvCharMethod(BSTR* pValue)
HRESULT put_RcvCharMethod(BSTR pNewValue)

Interface INoiseCal3

Last Modified:

9-Jun-2011 MN New topic

R1InputPath Property

Description All PNA models (except the N523xA models) have a switch in the test set that allows access to the port 1 reference receiver through the front panel Reference 1 connectors. This command throws that switch between the internal path to the receiver, or through the external connectors.

See other [Frequency Offset](#) properties.

VB Syntax `chan.R1InputPath = value`

Variable [\(Type\) - Description](#)

chan A [Channel](#) (object)

value **(Enum as naInputPath)** - Choose from: naPathInternal -

(0) - internal path to the reference receiver naPathExternal

(1) - path through external connectors

Return Type Enum

Default naPathInternal - (0)

Examples `chan.R1InputPath = naPathInternal 'Write`

`Inpath = chan.R1InputPath 'Read`

C++ Syntax `HRESULT get_R1InputPath (tag NAInputPath *pPath);`

`HRESULT put_R1InputPath (tag NAInputPath newPath);`

Interface IChannel2

Last Modified:

30-Apr-2009 Added PNA-X models

ReadingsPerPoint Property

Description This command, along with [ReadingsTolerance](#), allows for settling of the power sensor READINGS.

Specifies the maximum number of power readings that are taken at each stimulus point to allow for power meter settling. Each reading is averaged with the previous readings at that stimulus point. When this average meets the [ReadingsTolerance](#) value or this number of readings has been made, the average is returned as the valid reading.

VB Syntax *pwrSensor*.**ReadingsPerPoint** = *value*

Variable [\(Type\)](#) - **Description**

pwrCal A [SourcePowerCalibrator](#) (Object) or

A [PowerSensorAsReceiver](#) (Object)

value **(long integer)** – Number of power readings. Choose any number between 3 and 100.

Return Type Long Integer

Default 3

Examples

```
pwrSensor.ReadingsPerPoint = 3 'Write  
numReadings = pwrSensor.ReadingsPerPoint 'Read
```

C++ Syntax HRESULT put_ReadingsPerPoint(long newVal);

HRESULT get_ReadingsPerPoint(long *pVal);

Interface ISourcePowerCalibrator

IPowerSensorAsReceiver

Last Modified:

25-Aug-2009 Added PMAR

17-Apr-2007 Clarified verbage

ReadingsTolerance Property

Description This command, along with [ReadingsPerPoint Property](#) allows for settling of the power sensor READINGS.

Each power reading is averaged with the previous readings at each stimulus point. When the average meets this tolerance value or the maximum [ReadingsPerPoint](#) has been made, the average is returned as the valid reading.

VB Syntax *pwrSens.ReadingsTolerance = value*

Variable [\(Type\) - Description](#)

pwrCal A [SourcePowerCalibrator](#) (Object) or

A [PowerSensorAsReceiver](#) (Object)

value **(Double)** – Power meter settling tolerance value in dB. Choose any number between 0 and 5.

Return Type Double

Default .05 dB

Examples

```
pwrSens.ReadingsTolerance = .1 'Write
ReadTol = pwrSensor.ReadingsTolerance 'Read
```

C++ Syntax HRESULT get_ReadingsTolerance(double *pVal);
HRESULT put_ReadingsTolerance(double newVal);

Interface ISourcePowerCalibrator3
IPowerSensorAsReceiver

Last Modified:

25-Aug-2009 Added PMAR
17-Apr-2007 Clarified verbage

ReadyForTriggerPolarity Property

Description Specifies the polarity of Ready for Trigger output.

All existing Ready for Trigger outputs for PNA-X and PNA-L models are configured simultaneously with this command. [See Capabilities Summary.](#)

The Ready for Trigger polarity can NOT be configured for E836x models.

VB Syntax *trigsetup*.**ReadyForTriggerPolarity** = *value*

Variable [\(Type\)](#) - **Description**

trigsetup A [TriggerSetup](#) (**object**)

value (Enum as NAlevel)

Choose from:

0 - naLow - Outputs a TTL low when the PNA is ready for trigger.

1 - naHigh - Outputs a TTL high when the PNA is ready for trigger.

Return Type Enum

Default **0** - naLow

Examples `trigsetup.ReadyForTriggerPolarity = naLow 'Write`

`pol = trigsetup.ReadyForTriggerPolarity 'Read`

C++ Syntax HRESULT get_ReadyForTriggerPolarity(tagNAlevel *pVal);

HRESULT put_ReadyForTriggerPolarity(tagNAlevel newVal);

Interface ITriggerSetup3

Last Modified:

14-Mar-2008 MX New topic

ReadyForTriggerState Property

Description Determines the control of Material Handler connector Pin 21.

VB Syntax *handler.ReadyForTriggerState = value*

Variable [\(Type\)](#) - Description

handler **(object)** - A [Handler I/O](#) object

value **(boolean)**

False - Pin 21 is controlled by Output Port B7

True - Pin 21 is controlled by the Ready for Trigger signal

Return Type Boolean

Default False

Examples `handler.ReadyForTriggerState = False 'Write
bState = handler.ReadyForTriggerState 'Read`

C++ Syntax HRESULT put_ReadyForTriggerState (BOOL *pVal);

HRESULT get_ReadyForTriggerState (BOOL newVal);

Interface IHWMaterialHandlerIO2

RecallSoftkeysMostRecent Property

Description Set and return whether to list files for recall on softkeys by most-recently used or alphabetically.

VB Syntax *pref.RecallSoftkeysMostRecent = value*

Variable [\(Type\)](#) - Description

pref A [Preferences](#) (**object**)

value (Boolean) – Choose from:

True – Recall softkeys show most recently-used files.

False – Recall softkeys show alphabetically-ordered files.

Return Type Boolean

Default False

Examples

```
pref.RecallSoftkeysMostRecent = False 'Write
```

```
recallPref = pref.RecallSoftkeysMostRecent 'Read
```

C++ Syntax HRESULT get_RecallSoftkeysMostRecent(VARIANT_BOOL * pVal);
HRESULT put_RecallSoftkeysMostRecent(VARIANT_BOOL pVal);

Interface IPreferences13

Last Modified:

22-Jul-2011 New topic (9.50)

ReceiverAttenuator Property

Description	Sets or returns the value of the specified receiver attenuator control.
VB Syntax	<i>object</i> . ReceiverAttenuator (<i>rec</i>) = <i>value</i>
Variable	(Type) - Description
<i>object</i>	Channel (object)
	or
	CalSet (object) - Read-only property
<i>rec</i>	(long integer) - Receiver with attenuator control to be changed. Choose from any of the available receivers in your PNA
	1 - Receiver A
	2 - Receiver B
	Receiver attenuation can not be set using logical receiver notation .
<i>value</i>	(double) - Attenuator value in dB. Choose any Long Integer between 0 and 35 in 5dB steps: If an invalid value is entered, the analyzer will select the next lower valid value. For example, if 19.9 is entered the analyzer will select 15 dB attenuation.
Return Type	Double
Default	0 db
Examples	<pre>chan.ReceiverAttenuator(1) = 5 'Write</pre> <pre>attn = chan.ReceiverAttenuator(2) 'Read</pre>
C++ Syntax	HRESULT get_ReceiverAttenuator(long lport, double *pVal) HRESULT put_ReceiverAttenuator(long lport, double newVal)
Interface	IChannel CalSet3

ReceiverAttenuator (Cal All) Property

Description	Sets and returns the Receiver Attenuator setting for a Cal All calibration.
VB Syntax	<i>calAll</i> . ReceiverAttenuator (<i>port</i>) = <i>value</i>
Variable	(Type) - Description
<i>calAll</i>	A CalibrateAllChannels (object)
<i>port</i>	(Long) Receiver port number.
<i>value</i>	(Double) Attenuation value in dB for a Cal All calibration. Choose a valid value for the PNA model. See valid settings .
Return Type	Double
Default	0 (zero dB)
Examples	<pre>calAll.ReceiverAttenuator = 0 'Set value</pre> <pre>value = calAll.ReceiverAttenuator 'Return value</pre>
C++ Syntax	HRESULT get_ReceiverAttenuator (long port, double val); HRESULT put_ReceiverAttenuator long port, double* newVal);
Interface	ICalibrateAllChannels

Last modified:

4-Jan-2012 New topic

ReceiverCount Property

Description	Returns the number of receivers in the remote PNA. The returned number includes both test port receivers and reference receivers. See the number of reference receivers in your PNA.
VB Syntax	<i>value</i> = <i>cap</i> . ReceiverCount
Variable	(Type) - Description
<i>value</i>	(Long) - Variable to store the returned number of receivers.
<i>cap</i>	A Capabilities (object)
Return Type	Long
Default	Not Applicable
Examples	<pre>value = cap.ReceiverCount 'Read</pre>
C++ Syntax	HRESULT get_ReceiverCount(long * receiverCount);
Interface	ICapabilities

ReceiverRatio Property

Description	Sets and returns the receiver ratio pair to be used with Receiver Leveling. To perform receiver leveling with a ratioed receiver, use ReceiverRatio Property .
VB Syntax	<i>RxLevel.ReceiverRatio(srcPort) = value</i>
Variable	(Type) - Description
<i>RxLevel</i>	A ReceiverLeveling Object
<i>srcPort</i>	(Long Integer) Source port for which to set the receiver for Receiver Leveling. Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use chan.getPortNumber to translate the string into a port number. To learn more see Remotely Specifying a Source Port
<i>value</i>	(String) Receiver to use for the leveling sweeps. (Not case sensitive). Choose any receiver in your PNA. see the block diagram of your PNA, located at the bottom of all Specs documents . Receivers can also be referred to using logical receiver notation. This notation makes it easy to refer to receivers with an external test set connected to the PNA. You do not need to know which physical receiver is used for each test port. Learn more .
Return Type	String
Default	Not applicable
Examples	<pre>rxLevel.ReceiverRatio (1) = "R1" ' Write</pre> <pre>rxLevel.ReceiverRatio (1) = "b2" ' Write</pre> <pre>value = rxLevel.ReceiverRatio 2 ' Read</pre>
C++ Syntax	<pre>HRESULT get_ReceiverRatio(long port, BSTR* pVal);</pre> <pre>HRESULT put_ReceiverRatio(long port, BSTR newVal);</pre>
Interface	IRceiverLevelingConfiguration

Last Modified:

22-Dec-2010 Added link to ratioed rec.
13-Feb-2009 MX New topic

ReceiverStepAttenuatorStepSize Property

Description Returns a value indicating the step size of the attenuator.

VB Syntax *value* = *cap*.**ReceiverStepAttenuatorStepSize** (*n*)

Variable [\(Type\)](#) - Description

value **(Double)** - Variable to store the returned value of the attenuator step size.

cap A [Capabilities](#) **(object)**

n **(Long)** - port number to query for the value of the attenuator step size.

Return Type Double

Default Not Applicable

Examples `value = cap.ReceiverStepAttenuatorStepSize(1) 'Read`

C++ Syntax HRESULT get_ReceiverStepAttenuatorStepSize(long portNumber, double * stepSize);

Interface ICapabilities

ReceiverTemperature Property

Description	Returns the current temperature on the PNA receiver board.
VB Syntax	<i>value</i> = <i>cap</i> . ReceiverTemperature (<i>unit</i>)
Variable	(Type) - Description
<i>value</i>	(Double) - Variable to store the returned temperature value.
<i>cap</i>	A Capabilities (object)
<i>unit</i>	(Enum as NATempScale) - Units in which temperature is returned. Choose from: <ul style="list-style-type: none">• 0 - naTempScale_Fahrenheit• 1 - naTempScale_Celsius
Return Type	Double
Default	Not Applicable
Examples	<pre>value = cap.ReceiverTemperature naTempScale_Celsius 'Read</pre>
C++ Syntax	HRESULT get_ReceiverTemperature(NATempScale val, long temperature);
Interface	ICapabilities9

Last modified:

22-Sep-2011 New topic

Read-only

ReceivePort Property

Description Returns the receiver (response) port number of measurement. To understand how this property is useful, see [IMeasurement2 Interface](#).

Note: Returning a receiver port is only supported for S-Parameter measurements. If the measurement is not an S-Parameter, then E_NA_BAD_PARAMETER is returned.

VB Syntax *value* = *meas*.**ReceivePort**

Variable [\(Type\)](#) - Description

value **(Long)** - Variable to store the returned value

meas A [Measurement](#) **(object)**

Return Type Long Integer

Default Not Applicable

Examples `rp = meas.ReceivePort`

C++ Syntax HRESULT ReceivePort(Long* rcvPort);

Interface IMeasurement2

RedTraceOnFail Property

Description Set and return whether to display limit line failures as red trace segments or red data points (dots).

VB Syntax *pref.RedTraceOnFail = bool*

Variable [\(Type\)](#) - Description

pref A [Preferences](#) (**object**)

bool (**Boolean**) - Choose from:

False - Display failures as red data points (dots).

True - Display failures as red trace segments. (Red Trace On Fail).

Return Type Boolean

Default False

Examples `pref.RedTraceOnFail = False 'Write`

`prefer = pref.RedTraceOnFail 'Read`

C++ Syntax HRESULT put_RedTraceOnFail(VARIANT_BOOL bValue)

HRESULT get_RedTraceOnFail(VARIANT_BOOL *bValue)

Interface IPreferences10

Last Modified:

11-Aug-2009 MX New topic

ReduceIFBandwidth Property

Description Sets or returns the state of the [Reduced IF Bandwidth at Low Frequencies](#) setting.

VB Syntax `chan.ReduceIFBandwidth = state`

Variable [\(Type\)](#) - Description

chan A [Channel](#) (object)

state (boolean)

False - Turns Reduce IFBW **OFF**

True - Turns Reduce IFBW **ON**

Return Type Boolean

Default True

Examples `chan.ReduceIFBandwidth = False 'Write`

`reduce = chan.ReduceIFBandwidth 'Read`

C++ Syntax HRESULT get_ReduceIFBandwidth(BOOL *pVal)
HRESULT put_ReduceIFBandwidth(BOOL newVal)

Interface IChannel5

Last Modified:

16-Aug-2007 Corrected Interface number

ReferenceCalFactor Property

Description	Sets and returns the Cal Factor (%) for the 50 MHz reference signal associated with this power sensor. Use this property only if the power sensor does not contain cal factors in EPROM (for example, HP/Agilent 848x sensors).
VB Syntax	<i>pwrSensor</i> . ReferenceCalFactor = <i>value</i>
Variable	(Type) - Description
<i>pwrSensor</i>	A PowerSensor (Object)
	A PowerSensorAsReceiver (Object)
<i>value</i>	(double) – Cal factor in units of percent. This can be any value between 1 and 150.
Return Type	Double
Default	100
Examples	<pre>pwrSensor.ReferenceCalFactor = 98 'R</pre> <pre>RefFact = pwrSensor.ReferenceCalFactor 'Read</pre>
C++ Syntax	HRESULT put_ReferenceCalFactor(double newVal); HRESULT get_ReferenceCalFactor(double *pVal);
Interface	IPowerSensor IPowerSensorAsReceiver

Last Modified:

25-Aug-2009 Added PMAR

ReferenceMarkerState Property

Description Turn ON or OFF the reference marker.

VB Syntax *meas*.**ReferenceMarkerState** = *state*

Variable [\(Type\)](#) - Description

app A Measurement (**object**)

state (boolean) -

True - turns the reference marker ON

False - turns the reference marker OFF

Return Type Boolean

Default **False**

Examples `meas.ReferenceMarkerState = True`

`reference = meas.ReferenceMarkerState`

C++ Syntax HRESULT get_ReferenceMarkerState(VARIANT_BOOL bState)
HRESULT put_ReferenceMarkerState(VARIANT_BOOL* bState)

Interface IMeasurement

ReferenceReceiver Property

Description	Sets and returns the receiver to be used with Receiver Leveling. To perform receiver leveling with a ratioed receiver, use ReceiverRatio Property .
VB Syntax	<i>RxLevel</i> . ReferenceReceiver (<i>srcPort</i>) = <i>value</i>
Variable	(Type) - Description
<i>RxLevel</i>	A ReceiverLeveling Object
<i>srcPort</i>	(Long Integer) Source port for which to set the receiver for Receiver Leveling. Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use chan.getPortNumber to translate the string into a port number. To learn more see Remotely Specifying a Source Port
<i>value</i>	(String) Receiver to use for the leveling sweeps. (Not case sensitive). Choose any receiver in your PNA. see the block diagram of your PNA, located at the bottom of all Specs documents . Receivers can also be referred to using logical receiver notation. This notation makes it easy to refer to receivers with an external test set connected to the PNA. You do not need to know which physical receiver is used for each test port. Learn more .
Return Type	String
Default	Not applicable
Examples	<pre>rxLevel.ReferenceReceiver (1) = "R1" ' Write</pre> <pre>rxLevel.ReferenceReceiver (1) = "b2" ' Write</pre> <pre>value = rxLevel.ReferenceReceiver 2 ' Read</pre>
C++ Syntax	HRESULT get_ReferenceReceiver(long port, BSTR* pVal); HRESULT put_ReferenceReceiver(long port, BSTR newVal);
Interface	IRceiverLevelingConfiguration

Last Modified:

22-Dec-2010 Added link to ratioed rec.
13-Feb-2009 MX New topic

ReferenceValue Property

Description	Sets or returns the value of the Y-axis Reference Level of the active trace.
VB Syntax	<i>trce</i> .ReferenceValue = <i>value</i>
Variable	(Type) - Description
<i>trce</i>	A Trace (object)
<i>value</i>	(double) - Reference Value. Units and range depend on the current data format.
Return Type	Double
Default	Not applicable
Examples	<pre>meas.ReferenceValue = 0 'Write</pre> <pre>rlev = meas.ReferenceValue 'Read</pre>
C++ Syntax	HRESULT get_ReferenceValue(double *pVal) HRESULT put_ReferenceValue(double newVal)
Interface	ITrace

ReferencePosition Property

Description Sets or returns the Reference Position of the active trace.

VB Syntax *trce*.ReferencePosition = *value*

Variable [\(Type\)](#) - Description

trce A Trace (**object**)

value (**double**) - Reference position on the screen measured in horizontal graticules from the bottom of the screen. Choose from any number between: **0** and **10**.

Return Type Double

Default 0

Examples `meas.ReferencePosition = 5 'Middle of the screen -Write`

`rpos = meas.ReferencePosition -Read`

C++ Syntax HRESULT get_ReferencePosition(double *pVal)
HRESULT put_ReferencePosition(double newVal)

Interface ITrace

RemoteCalStoragePreference Property

Description Specifies the default manner in which calibrations performed using COM or SCPI are to be stored.

Cal data is always stored to the channel's Cal Register regardless of this setting.

This setting survives instrument preset and reboot. It remains until changed by another invocation of this property.

VB Syntax `pref.RemoteCalStoragePreference = value`

Variable [\(Type\)](#) - [Description](#)

cal A [Preferences](#) (object)

value **(Enum)** - Choose from:

0 - naPreferCalRegister - Cal is saved ONLY to the channel's Cal Register.

1 - naPreferNewUserCalSet - Cal is automatically saved to a new User Cal Set file when performing a calibration using COM. The Cal Set name is automatically generated. This corresponds to pre-6.0 behavior. Use the [Name](#) property to change the name after the cal is complete.

2 - naPreferReuseCurrentCalSet - The cal is saved to the Cal Set is that is currently selected on the specific channel. This could be the channel's Cal Register. If the channel does not yet have a selected Cal Set, the cal will be saved to a new User Cal Set with an automatically-generated name.

Return Type Enum

Default 0 - naPreferCalRegister

Examples

```
pref.RemoteCalStoragePreference = naPreferNewUserCalSet 'Write
calStorageMode = pref.RemoteCalStoragePreference ' Read
```

C++ Syntax HRESULT get_RemoteCalStoragePreference(enum NARemoteCalStoragePreference* preference);

HRESULT put_RemoteCalStoragePreference(enum NARemoteCalStoragePreference val);

Interface IPreferences7

16-Apr-2007 MX New topic

ReportReceiverOverload Property

Description Set and return whether to display receiver overload warnings.

VB Syntax *pref*.**ReportReceiverOverload** = *value*

Variable [\(Type\)](#) - Description

pref A [Preferences](#) (object)

value (**Boolean**) - Choose from:

True Display overload warnings.

False Do NOT display overload warnings.

Return Type Boolean

Default True

Examples `pref.ReportReceiverOverload = True 'Write`

`value = pref.ReportReceiverOverload 'Read`

C++ Syntax HRESULT get_ReportReceiverOverload (VARIANT_BOOL PowerSweepRetraceMode* preference);
HRESULT put_ReportReceiverOverload (VARIANT_BOOL PowerSweepRetraceMode val)

Interface IPreferences12

Last Modified:

30-Aug-2010 MX New topic

ResBWList Property

Description Returns a list of Res BW values that are supported by the IM Spectrum apps.

VB Syntax *value* = *cap*.ResBWList

Variable [\(Type\)](#) - Description

value (Variant) - Variable to store the returned array of Res BW values

cap A [Capabilities](#) (Object)

Return Type Variant array

Default Not Applicable

Examples `'Read the supported IFBW values`

```
Set app = CreateObject("AgilentPNA835x.Application")
Set cap = app.Capabilities
list=cap.ResBWList
For i = 0 To UBound(list)
    msg = msg & list(i) & vbCrLf
Next
MsgBox msg
```

C++ Syntax HRESULT get_ResBWList(Variant *value);

Interface ICapabilities8

Last Modified:

23-May-2011 MX New topic

ResolutionBW Property

Description Sets and returns the Resolution Bandwidth for the IM Spectrum measurement.

VB Syntax *ims.ResolutionBW = value*

Variable [\(Type\)](#) - Description

ims An [IMSpectrum](#) Object

value (Double) Resolution BW in Hz. Choose from:

60k | 100k | 150k | 300k, 600k | 1.0M | 3.0M

If an invalid number is specified, the PNA will round up to the closest valid number.

Return Type Double

Default 600 kHz

Examples `ims.ResolutionBW = 150e3 'Write`

`value = ims.ResolutionBW 'Read`

C++ Syntax HRESULT get_ResolutionBW(double *pVal)
HRESULT put_ResolutionBW(double newVal)

Interface IIMSpectrum

Last Modified:

19-Aug-2008 MX New topic

Reverse2PortAdapter Property

Description Set and read whether or not to reverse ports on a 2-port fixture or adapter to be de-embedded.

VB Syntax *fixture.Reverse2PortAdapter (port) = bool*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

port PNA port number for which SNP file is to be de-embedded.

bool **True** - Reverse ports.

False - Do NOT reverse ports.

Return Type Boolean

Default False

Examples `fixture.Reverse2PortAdapter = True`

`value = fixture.Reverse2PortAdapter 'Read`

C++ Syntax HRESULT get_Reverse2PortAdapter(short portNum, VARIANT_BOOL *pRev);

HRESULT put_Reverse2PortAdapter(short portNum, VARIANT_BOOL bRev);

Interface IFixturing6

Last Modified:

16-Nov-2010 MX New topic

ReverseLinearPowerLevel Property

Description Set and read the reverse power level to the DUT. This is applied to the DUT output port when making reverse measurements like S22.

VB Syntax `gca.ReverseLinearPowerLevel = value`

Variable [\(Type\)](#) - Description

gca A [GainCompression](#) (object)

value (**double**) Reverse power level in dBm. Choose a value from +30 to (-30).

Return Type Double

Default -5

Examples

```
gca.ReverseLinearPowerLevel = -10 'Write
```

```
LinPwr = gca.ReverseLinearPowerLevel 'Read
```

C++ Syntax HRESULT get_ReverseLinearPowerLevel(double* pVal)

HRESULT put_ReverseLinearPowerLevel(double newVal)

Interface IGainCompression

Last Modified:

21-Nov-2007 MX New topic

RFOffOnReceiverOverload Property

Description Set and return whether to turn source power OFF when a receiver is overloaded.

VB Syntax *pref*.**RFOffOnReceiverOverload** = *value*

Variable [\(Type\)](#) - Description

pref A [Preferences](#) (object)

value (**Boolean**) - Choose from:

True Turn OFF source power to ALL ports when a receiver is overloaded.

False Power remains ON when a receiver is overloaded.

Return Type Boolean

Default False

Examples `pref.RFOffOnReceiverOverload = True` 'Write

`value = pref.RFOffOnReceiverOverload` 'Read

C++ Syntax HRESULT get_RFOffOnReceiverOverload (VARIANT_BOOL* preference);
HRESULT put_RFOffOnReceiverOverload (VARIANT_BOOL val)

Interface IPreferences12

Last Modified:

30-Aug-2010 MX New topic

RoleDevice Property

Description	This command replaces AssignSourceToRole Method and GetSourceByRole Method . Set and return the source to be used in the specified role. For example, use this command to set a source name to be used as the RF2 tone for a Swept IMD channel.
VB Syntax	<i>chan.RoleDevice(role) = source</i>
Variable	(Type) - Description
<i>chan</i>	A Channel (object)
<i>role</i>	(String) Role of the source. Not context-sensitive. Use DefinedRoles to read the valid roles for the channel.
<i>source</i>	(String) Source name to be used in the specified role. Use extDevices.Items to read a list of configured sources.
Return Type	String
Default	Not Applicable
Examples	<pre>chan.RoleDevice("RF2") = "MyEsg" 'Write source = chan.RoleDevice("RF2") 'Read</pre>
C++ Syntax	<pre>HRESULT get_RoleDevice(BSTR role BSTR *source); HRESULT put_RoleDevice(BSTR role BSTR source);</pre>
Interface	IChannel22

Last modified:

12-Mar-2014 New topic

SafeMode Property

Description	Sets and returns the state of Safe Mode.
VB Syntax	<i>RxLevel</i> . SafeMode (<i>srcPort</i>) = <i>value</i>
Variable	(Type) - Description
<i>RxLevel</i>	A ReceiverLeveling Object
<i>value</i>	(Boolean) Choose from: True - Safe mode ON False - Safe mode OFF
<i>srcPort</i>	(Long Integer) Source port for which to set the Safe Mode state for Receiver Leveling. Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use chan.getPortNumber to translate the string into a port number. To learn more see Remotely Specifying a Source Port
Return Type	Variant Boolean
Default	False
Examples	<pre>rxLevel.SafeMode (1) = True ' Write value = rxLevel.SafeMode 2 ' Read</pre>
C++ Syntax	HRESULT get_SafeMode(long port, VARIANT_BOOL* pLevelingSafeMode); HRESULT put_SafeMode(long port, VARIANT_BOOL LevelingSafeMode);
Interface	IRceiverLevelingConfiguration

Last Modified:

13-Feb-2009 MX New topic

SafeSweepCoarsePowerAdjustment Property

Description Set and read the Safe Sweep COARSE power adjustment.

VB Syntax `gca.SafeSweepCoarsePowerAdjustment = value`

Variable [\(Type\)](#) - Description

gca A [GainCompression](#) (object)

value (**Double**) Coarse power adjustment setting in dBm. Choose a value from +30 to (-30).

Return Type Double

Default 3.0

Examples

```
gca.SafeSweepCoarsePowerAdjustment = 2.0 'Write
```

```
SSCourse = gca.SafeSweepCoarsePowerAdjustment 'Read
```

C++ Syntax HRESULT get_SafeSweepCoarsePowerAdjustment(double* value)
HRESULT put_SafeSweepCoarsePowerAdjustment(double value)

Interface IGainCompression

Last Modified:

1-Dec-2007 MX New topic

SafeSweepEnable Property

Description Set and read the (ON | OFF) state of Safe Sweep mode.

VB Syntax `gca.SafeSweepEnable = value`

Variable [\(Type\)](#) - Description

gca A [GainCompression](#) (object)

value (**Boolean**) Safe Sweep state. Choose from:

False - Disable Safe Sweep

True - Enable Safe Sweep

Return Type Boolean

Default False

Examples `gca.SafeSweepEnable = True 'Write`

`SSEnable = gca.SafeSweepEnable 'Read`

C++ Syntax HRESULT get_SafeSweepEnable(VARIANT_BOOL* value)

HRESULT put_SafeSweepEnable(VARIANT_BOOL value)

Interface IGainCompression

Last Modified:

1-Dec-2007 MX New topic

SafeSweepFinePowerAdjustment Property

Description Set and read the Safe Sweep FINE power adjustment.

VB Syntax `gca.SafeSweepFinePowerAdjustment = value`

Variable [\(Type\)](#) - Description

gca A [GainCompression](#) (object)

value **(Double)** Fine power adjustment setting in dBm. Choose a value from +30 to (-30).

Return Type Double

Default 1.0

Examples `gca.SafeSweepFinePowerAdjustment = 0.1 'Write`

`ssfine = gca.SafeSweepFinePowerAdjustment 'Read`

C++ Syntax HRESULT get_SafeSweepFinePowerAdjustment(double* value)

HRESULT put_SafeSweepFinePowerAdjustment(double value)

Interface IGainCompression

Last Modified:

1-Dec-2007 MX New topic

SafeSweepFineThreshold Property

Description Set and read the compression level at which Safe Sweep changes from the COARSE power adjustment to the FINE power adjustment.

VB Syntax `gca.SafeSweepFineThreshold = value`

Variable [\(Type\)](#) - Description

gca A [GainCompression](#) (object)

value (**Double**) Threshold setting in dBm. Choose a value from +30 to (-30).

Return Type Double

Default 0.5 dBm

Examples `gca.SafeSweepFineThreshold = 0.1` **Write**

`SSThresh = gca.SafeSweepFineThreshold` **Read**

C++ Syntax HRESULT get_SafeSweepFineThreshold(double* value)

HRESULT put_SafeSweepFineThreshold(double value)

Interface IGainCompression

Last Modified:

1-Dec-2007 MX New topic

SafeSweepMaximumLimit Property

Description When the PNA port that is connected to the DUT Output measures the specified value, the input power to the DUT is no longer incremented at that frequency.

VB Syntax `gca.SafeSweepMaximumLimit = value`

Variable [\(Type\)](#) - Description

gca A [GainCompression](#) (object)

value (**Double**) Maximum power level in dBm. Choose a value from -100 to +100.

Return Type Double

Default 100

Examples `gca.SafeSweepMaximumLimit = 23 'Write`

`maxPwr = gca.SafeSweepMaximumLimit 'Read`

C++ Syntax HRESULT get_SafeSweepMaximumLimit(double* value)

HRESULT put_SafeSweepMaximumLimit(double value)

Interface IGainCompression

Last Modified:

16-Mar-2010 MX New topic

SaturationLevel Property

Description Set and read the deviation dB from the maximum Pout. This is the point of saturation.

This value is used for [Compression Method](#): Compression from Saturation.

VB Syntax `gca.SaturationLevel = value`

Variable [\(Type\)](#) - Description

gca A [GainCompression](#) (object)

value **(Double)** - Saturation level in dB. Choose a value greater than 0.01 dB.

Return Type Double

Default .1 dB

Examples

```
gca.SaturationLevel = .5 'Write
```

```
satLevel = gca.SaturationLevel 'Read
```

C++ Syntax HRESULT get_SaturationLevel(double* pVal)

HRESULT put_SaturationLevel(double newVal)

Interface IGainCompression3

Last Modified:

3-Sep-2009 MX New topic

SB_BalPortNegative Property

Description With a Single-ended - Balanced topology, returns the PNA port number that is connected to the Negative side of the DUT's Balanced Port.

Use [SetSBPorts Method](#) to set the port mapping for a Single-Ended - Balanced topology.

VB Syntax *var* = *balTopology*.**SB_BalPortNegative**

Variable [\(Type\)](#) - Description

var (Long Integer) Variable to store the returned value.

balTopology A [BalancedTopology](#) (object)

Return Type Long Integer

Default Not Applicable

Examples `variable = balTop.SB_BalPortNegative` 'Read

C++ Syntax HRESULT get_SB_BalPortNegative(long *bVal)

Interface IBalancedTopology

Last modified:

Unknown New topic

SB_BalPortPositive Property

Description With a Single-ended - Balanced topology, returns the PNA port number that is connected to the Positive side of the DUT's Balanced Port.

Use [SetSBPorts Method](#) to set the port mapping for a Single-Ended - Balanced topology.

VB Syntax `var = balTopology.SB_BalPortPositive`

Variable [\(Type\)](#) - Description

`var` (Long Integer) Variable to store the returned value.

`balTopology` A [BalancedTopology](#) (object)

Return Type Long Integer

Default Not Applicable

Examples `variable = balTop.SB_BalPortPositive 'Read`

C++ Syntax HRESULT get_SB_BalPortPositive(long *bVal)

Interface IBalancedTopology

SB_SEPort Property

Description With a Single-ended - Balanced topology, returns the PNA port number that is connected to the DUT's Single-ended port.

Use [SetSBPorts Method](#) to set the port mapping for a Single-Ended - Balanced topology.

VB Syntax `var = balTopology.SB_SEPort`

Variable [\(Type\)](#) - Description

`var` (Long Integer) Variable to store the returned value.

`balTopology` A [BalancedTopology](#) (object)

Return Type Long Integer

Default Not Applicable

Examples `variable = balTopology.SB_SEPort 'Read`

C++ Syntax HRESULT get_SB_SEPort(long *bVal)

Interface IBalancedTopology

SBalMeasurement Property

Description Sets and returns the measurement for the Single-Ended - Balanced topology.

VB Syntax *balMeas.SBalMeasurement* = *value*

Variable [\(Type\)](#) - [Description](#)

balMeas A [BalancedMeasurement](#) (object)

value **(String)** - Single-ended - Balanced Measurement parameter. Not case-sensitive. Choose from:

Sss11	Ssd12	Ssc12
Sds21	Sdd22	Sdc22
Scs21	Scd22	Sc22
Imb	CMRR1	CMRR2
	(Sds21/Scs21)	(Ssd12/Ssc12)

Return Type Sss11

Default Not Applicable

Examples `balMeas.SBalMeasurement = "Ssd12" 'Write`

`variable = balMeas.SBalMeasurement 'Read`

C++ Syntax HRESULT get_SBalMeasurement(BSTR *pVal)

HRESULT put_SBalMeasurement(BSTR newVal)

Interface IBalancedMeasurement

ScaleCouplingMethod Property

Description	Sets and returns the method of scale coupling.
VB Syntax	<i>win</i> . ScaleCouplingMethod = <i>value</i>
Variable	(Type) - Description
<i>win</i>	An NAWindow (object)
	Any window object can be used to set this global property.
<i>value</i>	(enum NAScaleCouplingMethod)
	0 – naScaleCouplingOff - Scale Coupling is Off
	1 – naScaleCouplingWindow - Traces within selected windows share scaling
	2 – naScaleCouplingAll - Scaling is shared among traces in all selected windows
	Select windows using ScaleCouplingState Property
Return Type	Enum
Default	0 – naScaleCouplingOff
Examples	<pre>win.ScaleCouplingMethod = naScaleCouplingWindow 'Write</pre>
	<pre>method = app.ActiveNAWindow.ScaleCouplingMethod 'Read</pre>
C++ Syntax	HRESULT get_ScaleCouplingMethod(tagNAScaleCouplingMethod* couplingMethod);
	HRESULT put_ScaleCouplingMethod(tagNAScaleCouplingMethod couplingMethod);
Interface	INAWindow2

ScaleCouplingState Property

Description Enables and disables scale coupling for the window.
Use [ScaleCouplingMethod](#) to select the coupling method.

VB Syntax *win*.ScaleCouplingState = *bool*

Variable [\(Type\)](#) - Description

win An [NAWindow](#) (object) .

bool **(Boolean)**

False - NO scale coupling for this window.

True - Scale coupling enabled for this window.

Return Type Boolean

Default True

Examples

```
win.ScaleCouplingState = false 'Write
```

```
coupled = app.ActiveNAWindow.ScaleCouplingState  
'Read
```

C++ Syntax HRESULT get_ScaleCouplingState(VARIANT_BOOL
*pVal);

HRESULT put_ScaleCouplingState(VARIANT_BOOL
newVal)

Interface INAWindow2

Last Modified:

6-Aug-2009 MX New topic

Scope Property

Description Sets or returns the scope of a trigger signal. This determines whether a trigger signal affects a single channel or all channels in the PNA.

Note: [Trigger Modes](#) Point and EverySweep require that Trigger.Scope be set to naChannelTrigger.

VB Syntax `trigsetup.Scope = value`

Variable [\(Type\) - Description](#)

`trigsetup` A [TriggerSetup](#) (object)

`value` (enum **NATriggerType**) - Trigger type. Choose from:

0 - naGlobalTrigger - a trigger signal is applied to all triggerable channels

1 - naChannelTrigger - a trigger signal is applied to the current channel. The next trigger signal will be applied to the next channel; not necessarily the next channel in numeric sequence (1-2-3-4 and so forth).

Return Type Long Integer

Default naGlobalTrigger

Examples

```
trigsetup.Scope = naGlobalTrigger 'Write
trigtyp = trigsetup.Scope 'Read
```

C++ Syntax HRESULT get_Scope(tagNATriggerType *pTrigger)
HRESULT put_Scope(tagNATriggerType trigger)

Interface ITriggerSetup

Last Modified:

6-Nov-2007 Updated for new sweep mode

SearchFunction Property

Description Emulates the Tracking function in the marker search dialog box. The value you choose for SearchFunction will determine the type of search that takes place when the [Tracking](#) property is set true.

The tracking function finds the selected search function every sweep. In effect, turning Tracking ON is the same as executing one of the "Search..." methods (such as SearchMin, SearchMax) for every sweep.

VB Syntax *mark*.**SearchFunction** = *value*

Variable [\(Type\)](#) - Description

mark A [Marker](#) (object)

value (enum **NAMarkerFunction**) - search function. Choose from:

0 - naMarkerFunction_None

1 - naMarkerFunction_Min

2 - naMarkerFunction_Max

3 - naMarkerFunction_Target

4 - naMarkerFunction_NextPeak

5 - naMarkerFunction_PeakRight

6 - naMarkerFunction_PeakLeft

7 - naMarkerFunction_Compression

Return Type Long Integer

Default 0 - naMarkerFunction_None

Examples `mark.SearchFunction = naMarkerFunction_Target 'When this marker is set to track, it will track the Target value.`

`searchfunction = mark.SearchFunction 'Read`

C++ Syntax HRESULT get_SearchFunction(tagNAMarkerFunction *pVal)
HRESULT put_SearchFunction(tagNAMarkerFunction newVal)

Interface IMarker

SearchFailures Property

Description Returns a comma-separated list of the frequency indexes that were out of tolerance for SMART Sweep mode, or at the power limit for 2D Sweep mode. Zero (0) is the first frequency data point.

Must be Single triggered. Invalid results occur if the GCA channel is continuously sweeping.

VB Syntax *value* = *gca*.**SearchFailures**

Variable [\(Type\)](#) - [Description](#)

value **(Double)** Variable to store the returned data.

gca A [GainCompression](#) **(object)**

Return Type Returns a comma-separated list of frequency indexes.

Default Not applicable

Examples `SFA = gca.SearchFailures 'Read`

C++ Syntax HRESULT get_SearchFailures(VARIANT* value)

Interface IGainCompression

Last Modified:

11-Dec-2007 MX New topic

SearchSummary Property

Description Returns the status of a compression search.

This command can be used to indicate when a GCA search is complete.

Note: The returned value reflects the current state of the GCA compression search which can vary when in continuous sweep. This command is intended to be used with [chan.Single](#) (trigger).

VB Syntax *value* = *gca*.**SearchSummary**

Variable [\(Type\)](#) - Description

value **(Enum)** Variable to store the returned value.

- **0 - naSearchNotDone** - Acquisition is still in process.
- **1 - naSearchSucceeded** - Acquisition is complete and compression value found for all frequency points.
- **2 - naSearchFailed** - Acquisition is complete and unable to find the compression value at one or more frequency points.

gca A [GainCompression](#) **(object)**

Return Type Enum

Default Not Applicable

Examples `sum = gca.SearchSummary 'Read'`

C++ Syntax HRESULT get_SearchSummary(enum naGCASearchSummary* value)

Interface IGainCompression

Last Modified:

10-Jun-2009 MX New topic

SecurityLevel Property

Description Controls the display of frequency information on the PNA screen and printouts.

VB Syntax `app.SecurityLevel value`

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

value (enum `NASecurityLevel`) -Choose from:

0 - naNoSecurity ALL frequency information is displayed.

1 - naLowSecurity NO frequency information is displayed. Frequency information can be redisplayed using the Security Setting dialog box or this command.

2 - naHighSecurity LOW setting plus [GPIB console](#) is disabled. Frequency information can be redisplayed ONLY by performing a Preset, recalling an instrument state with None or Low security settings, or using this command.

3 - naExtraSecurity HIGH setting plus:

- [ASCII data saving](#) is disabled. Same method to redisplay frequency information as HIGH setting.
- [Mixer setup files](#) (*.mxr) can NOT be saved.

Return Type Long Integer

Default 0 - None

Examples

```
app.SecurityLevel = naLowSecurity 'Write
level = app.SecurityLevel 'Read
```

C++ Syntax `HRESULT get_NASecurityLevel(tagNASecurityLevel *level);`
`HRESULT put_NASecurityLevel(tagNASecurityLevel level);`

Interface `IApplication4`

Last Modified:

17-Jul-2007 Add Extra level

SegmentNumber Property

Description	Returns the number of the current segment, PowerSensorCalFactorSegment or PowerLossSegment object.
VB Syntax	<i>seg</i> .SegmentNumber
Variable	(Type) - Description
	<i>seg</i> Any of the following objects:
	A Segment (Object)
	A PowerSensorCalFactorSegment (Object)
	A PowerSensorCalFactorSegmentPMAR Object (Object)
	A PowerLossSegment (Object)
	A PowerLossSegmentPMAR (Object)
Return Type	Long Integer
Default	Not Applicable
Examples	<code>segNum = seg.SegmentNumber</code> 'returns the segment number -Read
C++ Syntax	HRESULT get_SegmentNumber(long *pVal)
Interface	ISegment IPowerSensorCalFactorSegment IPowerSensorCalFactorSegmentPMAR IPowerLossSegment IPowerLossSegmentPMAR

Last Modified:

27-Aug-2009 Added PMAR (9.0)

SegmentCount Property

Description	Returns the number of segments on the Applied mixer .
VB Syntax	<i>value</i> = <i>conv</i> . SegmentCount
Variable	(Type) - Description
<i>value</i>	(Long integer) Variable in which to store the returned segment count.
<i>conv</i>	A Converter Object
Return Type	Long integer
Default	1 Segment is created on new converter objects.
Examples	<pre>count=mxr.SegmentCount</pre> See example program
C++ Syntax	HRESULT get_SegmentCount(long *value);
Interface	IConverter5

Last Modified:

26-Oct-2010 New command (A.09.33)

SegmentFixedFrequency Property

Description	Set and read the CW Frequency for mixer segments. The specified SegmentRangeMode must be set to Fixed. Send Apply before sending a query (read). Learn more .
VB Syntax	<i>conv</i> . SegmentFixedFrequency (<i>index</i> , <i>range</i>) = <i>value</i>
Variable	(Type) - Description
<i>conv</i>	A Converter Object
<i>index</i>	(Long integer) Segment for which fixed frequency is being set. Choose a segment between 1 and the current segment count. Use SegmentCount Property to read the current count in the Applied Mixer .
<i>range</i>	(Enum as ConverterFrequencyRange) Range for which fixed frequency is being set. Choose from: 0 - naInputFrequencies - set input frequency 1 - naOutputFrequencies - set output frequency 2 - naLO1Frequencies - set LO1 frequency 3 - naLO2Frequencies - set LO2 frequency 4 - naIFFrequencies - set IF frequency
<i>value</i>	(Double) CW Frequency. Choose a value within the frequency range of the PNA.
Return Type	Double
Default	Center frequency of the PNA
Examples	<pre>mxr.SegmentFixedFrequency(1,0)=1e9 'sets the input frequency to 1 GHz</pre> See example program
C++ Syntax	HRESULT get_SegmentFixedFrequency(long index, tagConverterFrequencyRange range, double *value); HRESULT put_SegmentFixedFrequency(long index, tagConverterFrequencyRange value, double value);

Interface IConverter5

Last Modified:

26-Oct-2010 New command (A.09.33)

SegmentFixedPower Property

Description	Set and read the fixed power level for the mixer segment.
VB Syntax	<i>conv</i> . SegmentFixedPower (<i>index</i> , <i>range</i>) = <i>value</i>
Variable	(Type) - Description
<i>conv</i>	A Converter Object
<i>index</i>	(Long integer) Segment for which power is being set. Choose a segment between 1 and the current segment count. Use SegmentCount Property to read the current count in the Applied Mixer .
<i>range</i>	(Enum as ConverterFrequencyRange) Range for which power is being set. Choose from: <ul style="list-style-type: none"> 0 - naInputFrequencies - set input power 1 - naOutputFrequencies - set output power 2 - naLO1Frequencies - set LO1 power 3 - naLO2Frequencies - set LO2 power 4 - naIFFrequencies - set IF power
<i>value</i>	(Double) Power level in dBm. Choose a value within the power/attenuation range of the PNA.
Return Type	Double
Default	
Examples	<pre>mxr.SegmentFixedPower(1,0)=0 'sets the input power level to 0 dBm</pre> See example program
C++ Syntax	HRESULT get_SegmentFixedPower(long index, tagConverterFrequencyRange range, double *value); HRESULT put_SegmentFixedPower(long index, tagConverterFrequencyRange value, double value);
Interface	IConverter5

26-Oct-2010 New command (A.09.33)

SegmentIFBandwidth Property

Description	Sets and returns the IF Bandwidth for the sweep segment. Send Apply before sending a query (read). Learn more .
VB Syntax	<i>conv</i> . SegmentIFBandwidth (<i>index</i> , <i>value</i>)
Variable	(Type) - Description
<i>conv</i>	A Converter Object
<i>index</i>	(Long integer) Segment for which IF Bandwidth is to be set. Choose a segment between 1 and the current segment count. Use SegmentCount Property to read the current count in the Applied Mixer .
<i>value</i>	(Long integer) IF Bandwidth in Hz. The list of valid IF Bandwidths is different depending on the PNA model. See the lists . If an invalid number is specified, the analyzer will round up to the closest valid number.
Return Type	Long integer
Default	100 kHz
Examples	<pre>mxr.SegmentIFBandwidth(1,3)=10e3 'Sets IFBW to 10 kHz</pre> See example program
C++ Syntax	HRESULT get_SegmentIFBandwidth(long index, long * val); HRESULT put_SegmentIFBandwidth(long index, long val);
Interface	IConverter5

Last Modified:

26-Oct-2010 New command (A.09.33)

SegmentIsInputGreaterThanLO Property

Description Set and read whether to use the Input frequency that is greater than the LO or less than the LO.

Send [Apply](#) before sending a query (read). [Learn more](#).

VB Syntax *conv*.**SegmentIsInputGreaterThanLO**(*index*,*range*) = *value*

Variable [\(Type\)](#) - Description

conv A [Converter Object](#)

index (Long integer) Segment which is being set. Choose a segment between 1 and the current segment count. Use [SegmentCount Property](#) to read the current count in the [Applied Mixer](#).

range (Enum as ConverterFrequencyRange) Range for which value is being set. Choose from:

0 - naInputFrequencies - set input power

1 - naOutputFrequencies - set output power

2 - naLO1Frequencies - set LO1 power

3 - naLO2Frequencies - set LO2 power

4 - naIFFrequencies - set IF power

value (**Boolean**) -Choose from the following:

True - Use the Input that is Greater than the specified LO.

False - Use the Input that is Less than the specified LO.

Return Type Boolean

Default True

Examples

```
mxr.SegmentIsInputGreaterThanLO(1,2=1 'sets segment 1, LO1 to  
Input greater
```

[See example program](#)

C++ Syntax HRESULT get_SegmentIsInputGreaterThanLO(long index,
tagConverterFrequencyRange range, VARIANT_BOOL *value);

HRESULT put_SegmentIsInputGreaterThanLO(long index,

tagConverterFrequencyRange value, VARIANT_BOOL value);

Interface IConverter5

Last Modified:

26-Oct-2010 New command (A.09.33)

SegmentMixingMode Property

Description	Set and read whether the mixing mode (high side or low side) for the mixer segment. Send Apply before sending a query (read). Learn more .
VB Syntax	<i>conv</i> . SegmentMixingMode (<i>index</i> , <i>range</i>) = <i>value</i>
Variable	(Type) - Description
<i>conv</i>	A Converter Object
<i>index</i>	(Long integer) Segment for which mixing mode being set. Choose a segment between 1 and the current segment count. Use SegmentCount Property to read the current count in the Applied Mixer .
<i>range</i>	(Enum as ConverterFrequencyRange) Range for which mixing mode is being set. Choose from: 1 - naOutputFrequencies - sets output frequencies for 1-stage or 2-stage mixers. 4 - naIFFrequencies - sets IF frequencies for the first LO in 2-stage mixers.
<i>value</i>	(Enum as ConverterSideBand) Choose from: 0 - naLowSide Input minus LO 1 - naHighSide Input plus LO
Return Type	Enum as ConverterSideBand
Default	0 - naLowSide
Examples	<pre>mxr.SegmentMixingMode(1,1)=1 'sets segment 1 output frequencies to Highside mixing.</pre> See example program
C++ Syntax	HRESULT get_SegmentMixingMode(long index, tagConverterFrequencyRange range, tagConverterSideBand *value); HRESULT put_SegmentMixingMode(long index, tagConverterFrequencyRange value, tagConverterSideBand value);
Interface	ICConverter5

Last Modified:

26-Oct-2010 New command (A.09.33)

SegmentPoints Property

Description	Sets and returns the number of data points to be measured in the sweep segment. Send Apply before sending a query (read). Learn more .
VB Syntax	<i>conv</i> . SegmentPoints (<i>index</i>) = <i>value</i>
Variable	(Type) - Description
<i>conv</i>	A Converter Object
<i>index</i>	(Long integer) Segment for which points is to be set. Choose a segment between 1 and the current segment count. Use SegmentCount Property to read the current count in the Applied Mixer .
<i>value</i>	(Long integer) - Choose a value between 1 and the maximum number of data points allowed in the PNA . This is also the total number of points allowed for ALL segments.
Return Type	Long integer
Default	21
Examples	<pre>mxr.SegmentPoints(1)=3 'Sets 3 points for segment 1.</pre> See example program
C++ Syntax	HRESULT get_SegmentPoints(long index, long * val); HRESULT put_SegmentPoints(long index, long val);
Interface	IConverter5

Last Modified:

26-Oct-2010 New command (A.09.33)

SegmentRangeMode Property

Description	Sets or returns the segment sweep mode (Swept or Fixed) for the specified range (Input/LO/Output). Send Apply before sending a query (read). Learn more .
VB Syntax	<i>conv</i> . SegmentRangeMode (<i>index</i> , <i>range</i>) = <i>value</i>
Variable	(Type) - Description
<i>conv</i>	A Converter Object
<i>index</i>	(Long integer) Segment for which mixing mode being set. Choose a segment between 1 and the current segment count. Use SegmentCount Property to read the current count in the Applied Mixer .
<i>range</i>	(Enum as ConverterFrequencyRange) Range for which sweep mode is being set. Choose from: 0 - naInputFrequencies - set input sweep mode. 1 - naOutputFrequencies - set output sweep mode. 2 - naLO1Frequencies - set LO1 sweep mode. 3 - naLO2Frequencies - set LO2 sweep mode. 4 - naIFFrequencies - set IF sweep mode.
<i>value</i>	(Enum as NARangeMode) Choose from: 0 - naSwept Range is swept 1 - naFixed Range is fixed.
Return Type	Enum as NARangeMode
Default	Input and Output - naSwept LO (1 and 2) - naFixed
Examples	<pre>mxr.SegmentRangeMode(1,1)=0 'sets segment 1 output range to swept.</pre> See example program
C++ Syntax	HRESULT get_SegmentRangeMode(long index, tagConverterFrequencyRange range, tagNARangeMode *value);

```
HRESULT put_SegmentRangeMode(long index,  
tagConverterFrequencyRange value, tagNARangeMode value);
```

Interface IConverter5

Last Modified:

26-Oct-2010 New command (A.09.33)

SegmentStartFrequency Property

Description	Set and read the start frequency for the mixer segment. The specified SegmentRangeMode must be set to Swept. Send Apply before sending a query (read). Learn more .
VB Syntax	<i>conv</i> . SegmentStartFrequency (<i>index</i> , <i>range</i>) = <i>value</i>
Variable	(Type) - Description
<i>conv</i>	A Converter Object
<i>index</i>	(Long integer) Segment for which start frequency is being set. Choose a segment between 1 and the current segment count. Use SegmentCount Property to read the current count in the Applied Mixer .
<i>range</i>	(Enum as ConverterFrequencyRange) Range for which start frequency is being set. Choose from: 0 - naInputFrequencies - set input frequency 1 - naOutputFrequencies - set output frequency 2 - naLO1Frequencies - set LO1 frequency 3 - naLO2Frequencies - set LO2 frequency 4 - naIFFrequencies - set IF frequency
<i>value</i>	(Double) Start frequency. Choose a value within the frequency range of the PNA.
Return Type	Double
Default	Start frequency of the PNA
Examples	<pre>mxr.SegmentStartFrequency(1,0)=1e9 'sets the input start frequency to 1 GHz</pre> See example program
C++ Syntax	HRESULT get_SegmentStartFrequency(long index, tagConverterFrequencyRange range, double *value); HRESULT put_SegmentStartFrequency(long index, tagConverterFrequencyRange value, double value);

Interface IConverter5

Last Modified:

26-Oct-2010 New command (A.09.33)

SegmentState Property

Description Sets and returns the ON|OFF state of a sweep segment. Off segments are not included in a segment sweep.

Send [Apply](#) before sending a query (read). [Learn more](#).

VB Syntax *conv*.**SegmentState**(*index*) =*state*

Variable [\(Type\)](#) - Description

conv A [Converter Object](#)

index (Long integer) Segment to set ON or OFF. Choose a segment between 1 and the current segment count. Use [SegmentCount Property](#) to read the current count in the [Applied Mixer](#).

state (**Boolean**) - Choose from:

True - Segment ON

False - Segment OFF

Return Type Boolean

Default ON when added.

Examples

```
mxr.SegmentState(1)=False 'Turns segment 1 OFF.
```

[See example program](#)

C++ Syntax HRESULT get_SegmentState(long index, VARIANT_BOOL * val);

HRESULT put_SegmentState(long index, VARIANT_BOOL val);

Interface IConverter5

Last Modified:

26-Oct-2010 New command (A.09.33)

SegmentStopFrequency Property

Description	Set and read the Stop frequency for the mixer segment. The specified SegmentRangeMode must be set to Swept. Send Apply before sending a query (read). Learn more .
VB Syntax	<i>conv</i> . SegmentStopFrequency (<i>index</i> , <i>range</i>) = <i>value</i>
Variable	(Type) - Description
<i>conv</i>	A Converter Object
<i>index</i>	(Long integer) Segment for which Stop frequency is being set. Choose a segment between 1 and the current segment count. Use SegmentCount Property to read the current count in the Applied Mixer .
<i>range</i>	(Enum as ConverterFrequencyRange) Range for which Stop frequency is being set. Choose from: 0 - naInputFrequencies - set input frequency 1 - naOutputFrequencies - set output frequency 2 - naLO1Frequencies - set LO1 frequency 3 - naLO2Frequencies - set LO2 frequency 4 - naIFFrequencies - set IF frequency
<i>value</i>	(Double) Stop frequency. Choose a value within the frequency range of the PNA.
Return Type	Double
Default	Stop frequency of the PNA
Examples	<pre>mxr.SegmentStopFrequency(1,0)=1e9 'sets the input Stop frequency to 1 GHz</pre> See example program
C++ Syntax	HRESULT get_SegmentStopFrequency(long index, tagConverterFrequencyRange range, double *value); HRESULT put_SegmentStopFrequency(long index, tagConverterFrequencyRange value, double value);
Interface	IConverter5

Last Modified:

26-Oct-2010 New command (A.09.33)

SelectPort Property

Description Sets and returns a port mapping for a single port. If this command creates a conflict with an existing port, the PNA will resolve the conflict.

Note: This command is currently not supported for the Z5623AK44.

VB Syntax *tset.SelectPort(chNum, portNum) = portValue*

Variable [\(Type\) - Description](#)

tset A [TestsetControl](#) object.

chNum **(Long)** Channel number of the measurement.

portNum **(Long)** Physical port number to map.

portValue **(Long)** **Logical** port value to assign

Return Type Long

Default Not Applicable

Examples [See External Testset Program](#)

C++ Syntax HRESULT get_SelectPort(long channelNum, long PortNum long *outPort);
HRESULT put_SelectPort(long channelNum, long PortNum long outPort);

Interface ITestsetControl

SensorIndex Property

Description	For dual sensor power meters, sets and returns the power sensor channel (1 or 2) to be used.
VB Syntax	<i>pwrSensor</i> . SensorIndex = <i>value</i>
Variable	(Type) - Description
<i>pwrSensor</i>	A PowerSensorAsReceiver (Object)
<i>value</i>	(Long) - Power Meter channel. Choose from: 1 - Sensor A 2 - Sensor B
Return Type	Long
Default	1
Examples	<pre>pwrSensor.SensorIndex = False 'Write</pre> <pre>sensor = pwrSensor.SensorIndex 'Read</pre>
C++ Syntax	HRESULT put_SensorIndex(long newVal); HRESULT get_SensorIndex(long* pVal);
Interface	IPowerSensorAsReceiver

Last Modified:

25-Aug-2009 MX New topic

SeparatePowerCal Property

Description Specifies whether to use a Thru standard or to use two power sensor connections during the power cal of an SMC calibration. [Learn more.](#)

This command must be sent immediately after the Initialize command, but before all other calibration properties.

VB Syntax *smc*.**SeparatePowerCal** = *bool*

Variable [\(Type\)](#) - Description

smc An [SMCType](#) (object)

bool (Boolean)

True - Do NOT use a Thru, but instead perform separate power cals on Input and Output reference planes.

False - Perform Cal with Thru standard.

Return Type Boolean

Default False

Example

```
FCAppLib.ISMCType4 SMC =  
(FCAppLib.ISMCType4)CalMgr.CreateCustomCal("SMC");  
SMC.Initialize(chan, true);  
if (separatePowerCalIsDesired)  
SMC.SeparatePowerCal = true;
```

C++ Syntax HRESULT put_SeparatePowerCal(VARIANT_BOOL bValue);
HRESULT get_SeparatePowerCal(VARIANT_BOOL *bValue);

Interface SMCType4

Last Modified:

16-Sep-2009 MX New topic

ShowStatistics Property

Description Displays and hides the measurement (Trace) statistics (peak-to-peak, mean, standard deviation) on the screen. To display measurement statistics for a narrower band of the X-axis, use [StatisticsRange](#).

The analyzer will display either measurement statistics or Filter Bandwidth statistics; not both.

VB Syntax *meas*.**ShowStatistics** = *value*

Variable [\(Type\)](#) - **Description**

meas A Measurement (**object**)

value (**boolean**) - Boolean value:

True - Show statistics

False - Hide statistics

Return Type Boolean

Default **False**

Examples `meas.ShowStatistics = True`

C++ Syntax HRESULT put_ShowStatistics(VARIANT_BOOL bState)

Interface IMeasurement

Last modified:

12-Dec-2013 Removed Read

ShowProperties Property

Description	<p>Turns ON and OFF the display of the test set control status bar. This status bar indicates the test set that is being controlled and the current port mappings.</p> <p>This setting is turned ON and OFF automatically when the test set is enabled or disabled.</p>
VB Syntax	<i>tset</i> . ShowProperties = <i>value</i>
Variable	(Type) - Description
<i>tset</i>	A TestsetControl object.
	OR
	An E5091Testset object.
<i>value</i>	(Boolean)
	True - Turns display of testset properties ON.
	False - Turns display of testset properties OFF.
Return Type	Boolean
Default	False (True when test set control is enabled.)
Examples	See E5091A Example Program
	See External Testset Program
C++ Syntax	HRESULT get_ShowProperties(VARIANT_BOOL *state); HRESULT put_ShowProperties(VARIANT_BOOL state);
Interface	IE5091Testsets ITestsetControl

SICL Property

Description Allows you to control the PNA via SICL (standard instrument control library). In this mode, the analyzer can receive SCPI commands from the LAN interface or from a program residing on the PNA itself. This command performs the same function as the [SICL / GPIB](#) dialog box - **SICL Enabled** checkbox. [See Configuring the analyzer for SICL/VISA.](#)

When **SICL** is enabled, the PNA VXI-11.2 interface is enabled, and if the PNA hard disk image is new enough to have the VXI-11.3 interface, it also enables that. [Learn more about LXI / VXI.](#)

With this method you can augment a test program written using SICL that resides on the PNA so that it will run unattended. An automation script can be written to start the PNA, enable SICL (using the SICL property), and then start the SICL based program.

VB Syntax *app.SICL value*

Variable [\(Type\)](#) - **Description**

app An [Application](#) (**object**)

value (**Boolean**) Choose from:

True - enable SICL

False - disable SICL

Return Type Boolean

Default False

Examples

```
Dim Pna as AgilentPNA835x.Application
Dim siclState as Boolean
Set Pna = CreateObject("AgilentPNA835x.Application")
Pna.SICL = true           'write

siclState = Pna.SICL     'Read
```

C++ Syntax HRESULT get_SICL(VARIANT_BOOL *pVal)

HRESULT put_SICL(VARIANT_BOOL newVal)

Interface IApplication5

SICLAddress Property

Description Sets and returns the PNA SICL address. This is the address used for SICL over LAN.

VB Syntax *app.SICLAddress = value*

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

value (Integer) SICL Address of the PNA. Choose a value between 0 and 30.

Return Type Short Integer

Default 16

Examples `address=app.SICLAddress 'Read`

`app.SICLAddress=16 'Write`

C++ Syntax HRESULT get_SICLAddress(short busIndex, short* address);

HRESULT put_SICLAddress(short busIndex,short address);

Interface IApplication8

Simultaneous2PortAcquisition Property

Description Specifies whether a 2-port calibration will be done with a single set of standards (one port at a time) or with two sets of standards (simultaneously).

The [AcquireCalStandard2](#) command uses the same standard index for each calibration class. To specify the calibration standard gender for each port, you must first ensure that the order of calibration class accurately reflects the configuration of your DUT. For example, for a DUT with a male connector on port 1 and a female connector on port 2, order the devices within the S11 classes (A, B, and C) such that the MALE standards are first in the list. Then order the S22 classes specifying the FEMALE standards as the first in the list.

VB Syntax *cal*.**Simultaneous2PortAcquisition** = *state*

Variable [\(Type\)](#) - Description

cal A Calibrator (**object**)

state (**boolean**) - Choose from:

True - measures 2 ports simultaneously

False - measures 1 port at a time

Return Type Boolean

Default True

Examples `cal.Simultaneous2PortAcquisition = True`

C++ Syntax HRESULT put_Simultaneous2PortAcquisition(VARIANT_BOOL
bTwoSetsOfStandards)

HRESULT Simultaneous2PortAcquisition(VARIANT_BOOL
*bTwoSetsOfStandards)

Interface ICalibrator

Last modified:

9/20/06 Changed default to True

9/12/06 Modified for cross-browser

SmartSweepMaximumIterations Property

Description Set and read the maximum permitted number of iterations which SMART Sweep may utilize to find the desired compression level, to within the specified tolerance.

VB Syntax `gca.SmartSweepMaximumIterations = value`

Variable [\(Type\)](#) - Description

gca A [GainCompression](#) (object)

value (**integer**) - Maximum number of iterations. Choose a value between 1 and 50.

Return Type Integer

Default 20

Examples

```
gca.SmartSweepMaximumIterations = 10 'Write
```

```
iters = gca.SmartSweepMaximumIterations 'Read
```

C++ Syntax HRESULT get_SmartSweepMaximumIterations(int* pVal)

HRESULT put_SmartSweepMaximumIterations(int newVal)

Interface IGainCompression

Last Modified:

11-Sep-2007 MX New topic

SmartSweepSettlingTime Property

Description Set and read the amount of time SMART Sweep will dwell at the first point where the input power changes by the Backoff or X level.

[Learn more.](#)

VB Syntax `gca.SmartSweepSettlingTime = value`

Variable [\(Type\)](#) - Description

gca A [GainCompression](#) (object)

value (**double**) - Settling time in seconds. Choose any positive value.

Return Type Double

Default 0

Examples `gca.SmartSweepSettlingTime = .01 'Write`

```
sTime = gca.SmartSweepSettlingTime 'Read
```

C++ Syntax HRESULT get_SmartSweepSettlingTime(double* pVal)

HRESULT put_SmartSweepSettlingTime(double newVal)

Interface IGainCompression

Last Modified:

21-Nov-2007 MX New topic

SmartSweepShowIterations Property

Description Set and read whether to show intermediate results for each iteration in SMART sweep.

VB Syntax `gca.SmartSweepShowIterations = value`

Variable [\(Type\)](#) - Description

gca A [GainCompression](#) (object)

value (**Boolean**) Choose from:

True Compression traces are updated after each iteration.

False Compression traces are updated after ALL iterations are complete.

Return Type Boolean

Default False

Examples `gca.SmartSweepShowIterations = True 'Write`

`SShow = gca.SmartSweepShowIterations 'Read`

C++ Syntax HRESULT get_SmartSweepShowIterations(VARIANT_BOOL *pVal)

HRESULT put_SmartSweepShowIterations(VARIANT_BOOL newVal)

Interface IGainCompression

Last Modified:

21-Nov-2007 MX New topic

SmartSweepTolerance Property

Description Set and read the acceptable range SMART Sweep will allow for the measured compression level.

VB Syntax `gca.SmartSweepTolerance = value`

Variable [\(Type\)](#) - Description

`gca` A [GainCompression](#) (object)

`value` (**double**) - Tolerance level in dB. Choose a value between .01 and 10

Return Type Double

Default .05

Examples `gca.SmartSweepTolerance = .01 'Write`

`tol = gca.SmartSweepTolerance 'Read`

C++ Syntax HRESULT get_SmartSweepTolerance(double* pVal)

HRESULT put_SmartSweepTolerance(double newVal)

Interface IGainCompression

Last Modified:

11-Sep-2007 MX New topic

SmoothingAperture Property

Description Specifies or returns the amount of smoothing as a ratio of the number of data points in the measurement trace.

There is no COM command for specifying smoothing by number of aperture points.

VB Syntax *meas.SmoothingAperture = value*

Variable [\(Type\)](#) - Description

meas A [Measurement](#) (object)

value (**double**) - Smoothing Aperture. A ratio of (aperture points / trace points). Choose any number between **.01** and **.25**.

Return Type Double

Default .25

Examples

```
meas.SmoothingAperture = .10 'Write
saperture = meas.SmoothingAperture 'Read
```

C++ Syntax HRESULT get_SmoothingAperture(double *pVal)
HRESULT put_SmoothingAperture(double newVal)

Interface IMeasurement

Last modified:

Oct. 25, 2006 Fixed formula for smoothing

Smoothing Property

Description	Turns ON and OFF data smoothing.
VB Syntax	<i>meas.Smoothing</i> = <i>state</i>
Variable	(Type) - Description
<i>meas</i>	A Measurement (object)
<i>state</i>	(boolean) True - Turns smoothing ON False - Turns smoothing OFF
Return Type	Boolean
Default	False
Examples	<pre>meas.Smoothing = False 'Write</pre> <pre>smooth = meas.Smoothing 'Read</pre>
C++ Syntax	HRESULT get_Smoothing(VARIANT_BOOL *pVal) HRESULT put_Smoothing(VARIANT_BOOL newVal)
Interface	IMeasurement

SnPFormat Property

Description Specifies the format of .SnP files.

Use either app.[Save](#) (saves data to file) or meas.[Get SnpDataWithSpecifiedPorts](#) (reads data into variant array).

VB Syntax *pref.SnPFormat = value*

Variable [\(Type\)](#) - Description

pref A [Preferences](#) (object)

value (**string**) - Format of the .S1P, .S2P, .S3P, .S4P data. Choose from:

"MA" - Linear Magnitude / degrees

"DB" - Log Mag / degrees

"RI" - Real / Imaginary

"Auto" - Format in which the trace is already displayed. If other than Log Mag, Linear Magnitude, or Real/Imag, then the format will be in Real/Imag.

Return Type String

Default "Auto"

Examples

```
pref.SnPFormat = "MA" 'Write
```

```
format = pref.SnPFormat 'Read
```

C++ Syntax HRESULT get_SnPFormat(BSTR *Format)
HRESULT put_SnPFormat(BSTR Format)

Interface IPreferences

Last Modified:

13-Jun-2011 Updated list to get method

SoftwareGateState Property

Description When set to OFF, the improved software gating sensitivity is turned OFF and all data outside the measurement band is zeroed. This setting is used for troubleshooting purposes. There is NO user-interface control for this setting.

VB Syntax *pulseMeas*.**SoftwareGateState** = *bool*

Variable [\(Type\)](#) - **Description**

pulseMeas A [PulseMeasurementControl](#) (**object**)

bool **True** - Turn ON software gating.

False - Turn OFF software gating.

Return Type Boolean

Default True

Examples

```
pulse.SoftwareGateState = True 'Write  
value = pulse.SoftwareGateState 'Read
```

C++ Syntax HRESULT get_SoftwareGateState(VARIANT_BOOL *pVal);
HRESULT put_SoftwareGateState(VARIANT_BOOL newVal);

Interface IPulseMeasurementControl2

Last modified:

9-May-2011 New topic

SoundOnFail Property

Description Turns ON or OFF the audio indicator for limit failures.

VB Syntax *limitst*.SoundOnFail = *state*

Variable [\(Type\)](#) - Description

limitst A LimitTest (**object**)

state (**boolean**)

False - Turns the sound OFF

True - Turns the sound ON

Return Type Long Integer

Default **True**

Examples `Limttest.SoundOnFail = False 'Write`

`sound = Limttest.SoundOnFail 'Read`

C++ Syntax HRESULT get_SoundOnFail(VARIANT_BOOL *pVal)
HRESULT put_SoundOnFail(VARIANT_BOOL newVal)

Interface ILimitTest

Source Property

Description Sets or returns the source of triggering on the PNA and PNA-X.

VB Syntax *trigSetup.Source = value*

Variable [\(Type\)](#) - Description

trigSetup A [TriggerSetup](#) (object)

value (enum **NATriggerSource**) - Choose from:

0 - naTriggerSourceInternal - free run

1 - naTriggerSourceManual - manual trigger source; use app.[ManualTrigger](#) to send a trigger signal.

2 - naTriggerSourceExternal - a trigger signal is generated when a signal is sensed on the appropriate external trigger input connector. Use [ExternalTriggerConnectionBehavior](#) to configure the characteristics of the external trigger signal.

This setting has implications on Calibration. [Learn more.](#)

Return Type Long Integer

Default naTriggerSourceInternal

Examples

```
trigSetup.Source = naTriggerSourceInternal 'Write
```

```
trigsource = trigSetup.Source 'Read
```

C++ Syntax HRESULT get_Source(tagNATriggerSource *pTrigger);
HRESULT put_Source(tagNATriggerSource trigger);

Interface ITriggerSetup

Last Modified:

12-Dec-2011 Modified for PNA-X and N522x models

Sources Property

Description Returns the names of the configured DC sources for the specified channel.

VB Syntax *names* = *dc*.**Sources**

Variable [\(Type\)](#) - Description

names (Variant) Variable to store the returned DC source names.

dc An [DCStimulus](#) (**object**)

Default Not Applicable

Example `names = dc.Sources 'Read`

C++ Syntax HRESULT get_Sources(VARIANT * pValue);

Interface IDCStimulus

Last Modified:

21-Feb-2012 New topic

SourceAttenuator (Cal All) Property

Description	Sets and returns the Source Attenuator setting for a Cal All calibration.
VB Syntax	<i>calAll.SourceAttenuator (port) = value</i>
Variable	(Type) - Description
<i>calAll</i>	A CalibrateAllSourceAttenuator (object)
<i>port</i>	(Long) Source port number.
<i>value</i>	(Double) Attenuation value in dB for a Cal All calibration. Choose a valid value for the PNA model. See valid settings .
Return Type	Double
Default	0 (zero dB)
Examples	<pre>calAll.SourceAttenuator = 0 'Set value</pre> <pre>value = calAll.SourceAttenuator 'Return value</pre>
C++ Syntax	HRESULT get_SourceAttenuator (long port, double val); HRESULT put_SourceAttenuator long port, double* newVal);
Interface	ICalibrateAll

Last modified:

4-Jan-2012 New topic

SourceAttenuator Property

Description Sets and returns the Source Attenuator setting for the Phase Reference calibration.

Note: This setting MUST match the source attenuator setting at the mixer input port for subsequent SMC+Phase measurements.

VB Syntax *phasRef.SourceAttenuator* = *value*

Variable [\(Type\)](#) - [Description](#)

phasRef A [PhaseReferenceCalibration](#) Object

value Attenuation value in dB. Choose a valid value for the PNA model. [See valid settings](#).

Return Type Double

Default 10 dB

Examples `phase.SourceAttenuator = 0`

[See example program](#)

C++ Syntax HRESULT get_SourceAttenuator(Double* pVals);

HRESULT put_SourceAttenuator(Double pVals);

Interface IPhaseReferenceCalibration

Last Modified:

3-Apr-2012 MX New topic

SourceCount Property

Description Returns the number of sources in the remote PNA.

VB Syntax *value* = *cap*.**SourceCount**

Variable [\(Type\)](#) - Description

value (Long) - Variable to store the returned number of sources.

cap A [Capabilities](#) (**object**)

Return Type Long

Default Not Applicable

Examples `value = cap.SourceCount 'Read`

C++ Syntax HRESULT get_SourceCount(long * sourceCount);

Interface ICapabilities

SourceImpedance Property

Description Sets and returns the source impedance of the pulse generator.

VB Syntax *extPulseGen.SourceImpedance = value*

Variable [\(Type\)](#) - [Description](#)

extPulseGen An [ExternalPulseGenerator](#) (**object**)

value (**Double**) Pulse generator source impedance.

Return Type Double

Default **50**

Examples

```
extPulseGen.SourceImpedance = 50 'Write
```

```
srcImp = extPulseGen.SourceImpedance 'Read
```

C++ Syntax HRESULT get_SourceImpedance (double *pValue)

HRESULT put_SourceImpedance (double newVal)

Interface IExternalPulseGenerator

Last Modified:

15-Feb-2012 New topic

Read-only

SourcePort Property

Description	Returns the source port of measurement. To understand how this property is useful, see IMeasurement2 Interface .
VB Syntax	<i>value</i> = <i>meas</i> . SourcePort
Variable	(Type) - Description
<i>meas</i>	A Measurement (object)
<i>value</i>	(Long) - Variable to store the returned value
Return Type	Long Integer
Default	Not Applicable
Examples	<pre>sp = meas.SourcePort</pre>
C++ Syntax	HRESULT SourcePort([out, retval] Long* srcPort);
Interface	IMeasurement2

SourcePortCount Property

Description Returns the number of ports that can output a signal.

To learn more, see [Remotely Specifying a Source Port](#).

VB Syntax *value* = *object*.**SourcePortCount**

Variable [\(Type\)](#) - Description

value (Long) - Variable to store the returned integer value of the number of source ports.

object A [Channel](#) (**object**)- always more complete than capabilities object.

A [Capabilities](#) (**object**) - use when a channel is not available, or to find the common ports across all channels.

Return Type Long

Default Not Applicable

Examples `value = chan.SourcePortCount 'Read`

C++ Syntax HRESULT get_SourcePortCount(long * count);

Interface IChannel13
ICapabilities4

Last Modified:

23-May-2008 Added channel object

14-Jan-2007 MX New topic

SourcePortMode Property

Description Sets the state of the PNA source for the specified port.

VB Syntax *chan.SourcePortMode (sourcePort) = value*

Variable [\(Type\)](#) - Description

chan **(object)** - A [Channel](#) object

sourcePort **(long integer)** - The source port for which to make this setting.

Note: If the port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-x model, then you must use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#).

value **(enum)** - State of the source. Choose from:

0 - naSourcePortAuto Port power is turned on when required for a measurement.

1 - naSourcePortOn Port power is always ON, regardless of the measurement.

2 - naSourcePortOff Port power is always OFF, regardless of the measurement.

Return Type Enum

Default 0 - naSourcePortAuto

Examples

```
chan.SourcePortMode(1) = naSourcePortOn 'Write
state = chan.SourcePortMode(4) 'Read
```

C++ Syntax HRESULT get_SourcePortMode(long sourcePort, enum NASourcePortMode*);
HRESULT put_SourcePortMode(long sourcePort, enum NASourcePortMode);

Interface IChannel9

Last modified:

10-Mar-2011 Removed ON/OFF restrictions
24-Apr-2008 Added note for strings
30-Apr-2007 Edited for src strings
10/18/06 MQQ New topic

SourcePortNames Property

Description Returns the string names of ports that can output a signal.

The following is a list of string names for the PNA-X. Your PNA will NOT have all of these ports. Use [GetPortNumber Method](#) to return the correct port number for the specified port name.

- "Port 1"
- "Port 2"
- "Port 3"
- "Port 4"
- "Src2 Out1"
- "Src2 Out2"
- "Port 1 Src2"

For [iTMSA \(Opt 460\)](#)

- "Bal Port 1"
- "Bal Port 2"
- "SE Port1"
- "SE Port 2"

This command also lists the [External Sources](#) that are currently configured and selected.

To learn more, see [Remotely Specifying a Source Port](#).

VB Syntax *value* = *object*.**SourcePortNames**

Variable [\(Type\)](#) - **Description**

value (Variant array) - Variable to store the returned integer value of the number of source ports.

object A [Channel](#) (**object**) - always more complete than capabilities object.

A [Capabilities](#) (**object**) - use when a channel is not available, or to find the common ports across all channels.

Return Type Variant array of string names.

Default Not Applicable

Examples `value = chan.SourcePortNames 'Read`

C++ Syntax HRESULT get_SourcePortNames(VARIANT *names) ;

Interface IChannel13

ICapabilities4

Last Modified:

3-Mar-2009	Added SE Port 2
23-May-2008	Added iTMSA sources and channel object
23-Jul-2007	Clarification
14-Jan-2007	MX New topic

SourcePowerCalPowerOffset Property

Description Sets or returns a power level offset from the PNA test port power. This can be a gain or loss value (in dB) to account for components you connect between the source and the reference plane of your measurement. For example, specify 10 dB to account for a 10 dB amplifier at the input of your DUT.

Cal power is the sum of the test port power setting and this offset value. Following the calibration, the PNA power readouts are adjusted to the cal power.

This property performs the same function as the power offset argument on [SetCalInfoEx Method](#), except that this property can read the offset value.

VB Syntax `chan.SourcePowerCalPowerOffset (sourcePort) = value`

Variable [\(Type\)](#) - Description

chan **(object)** - A [Channel](#) object

sourcePort **(long integer)** - The source port for which to set this power offset value.

value **(double)** - Gain or loss value in dB. Choose a value between -200 and 200.

Return Type Double

Default 0 dB

Examples

```
chan.SourcePowerCalPowerOffset(1) = 10 'Write  
offset = chan.SourcePowerCalPowerOffset(2) 'Read
```

C++ Syntax HRESULT get_SourcePowerCalPowerOffset(long sourcePort, double *pVal);

HRESULT put_SourcePowerCalPowerOffset(long sourcePort, double newVal);

Interface IChannel4

Last Modified:

1-May-2007 Modified link to EX method.

SourcePowerCorrection Property

Description Sets source power correction ON or OFF for a specific source port on this channel, or returns the current ON or OFF state of correction for that source port.

VB Syntax `chan.SourcePowerCorrection (srcPort) = value`

Variable [\(Type\) - Description](#)

chan **(object)** – A Channel object

srcPort **(long integer)** – Source port for which to set or return the ON or OFF state of source power correction.

Note: If the port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-x model, then you must use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#).

value **(boolean)**

False – Turns source power correction OFF for the source port.

True – Turns source power correction ON for the source port.

Return Type Boolean

Default False - Source power correction will turn correction ON

Examples

```
chan.SourcePowerCorrection(1) = False 'Write
calOnPort2 = chan.SourcePowerCorrection(2) 'Read
```

C++ Syntax `HRESULT put_SourcePowerCorrection(VARIANT_BOOL bState);`
`HRESULT get_SourcePowerCorrection(VARIANT_BOOL *bState);`

Interface IChannel

Last Modified:

24-Apr-2008 Added note for string names

30-Apr-2007 Modified for src strings

SourcePowerOption Property

Description Enables the source power to be set on individual sweep segments. This property must be set **True** **before** `seg.TestPortPower = value` is sent. Otherwise, the test port power command will be ignored.

VB Syntax `segs.SourcePowerOption = state`

Variable [\(Type\)](#) - Description

segs A Segments collection (**object**)

state (**boolean**)

True - Enables variable TestPortPower to be set segment sweep

False - Disables variable TestPortPower to be set segment sweep

Return Type Boolean

Default False

Examples

```
segs.SourcePowerOption = True 'Write
```

```
powerOption = SourcePowerOption 'Read
```

C++ Syntax HRESULT get_SourcePowerOption(VARIANT_BOOL *pVal)
HRESULT put_SourcePowerOption(VARIANT_BOOL newVal)

Interface ISegments

SourcePowerState Property

Description Turns Source Power ON and OFF.

[See note about source power state with instrument state save and recall.](#)

VB Syntax `app.SourcePowerState = state`

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

state (boolean)

False - Turns Source Power OFF

True - Turns Source Power ON

Return Type Boolean

Default True

Examples `app.SourcePowerState = True 'Write`

`pwr = app.SourcePowerState 'Read`

C++ Syntax HRESULT get_SourcePowerState(VARIANT_BOOL *pVal)
HRESULT put_SourcePowerState(VARIANT_BOOL newVal)

Interface IApplication

SourcePullForSParameters Property

Description Enables and disables the use of source pull technique to compute S22 on Noise Figure on Converters. [Learn more.](#)

VB Syntax `nfx.SourcePullForSParameters = value`

Variable [\(Type\)](#) - Description

nfx A [NoiseFigure](#) (**object**)

value (Boolean) Source pull technique state. Choose from:

False - Disable use of source pull technique.

True - Enable use of source pull technique.

Return Type Boolean

Default **False**

Examples

```
nfx.SourcePullForSParameters = true 'Write
sourcePull = nfx.SourcePullForSParameters 'Read
```

C++ Syntax HRESULT get_SourcePullForSParameters (VARIANT_BOOL * Val)
HRESULT put_SourcePullForSParameters (VARIANT_BOOL newVal)

Interface INoiseFigure4

Last Modified:

7-Oct-2009 MX New topic

SourceStepAttenuatorStepSize Property

Description	Returns a value indicating the step size of the source attenuator.
VB Syntax	<i>value</i> = <i>cap</i> .SourceStepAttenuatorStepSize (<i>n</i>)
Variable	(Type) - Description
<i>value</i>	(Double) - Variable to store the returned value of the attenuator step size.
<i>cap</i>	A Capabilities (object)
<i>n</i>	(Long) - port number to query for the value of the attenuator step size.
Return Type	Double
Default	Not Applicable
Examples	<pre>value = cap.SourceStepAttenuatorStepSize(1)</pre>
C++ Syntax	HRESULT get_SourceStepAttenuatorStepSize(long portNumber, double * stepSize);
Interface	ICapabilities5

Last Modified:

30-Nov-2009 MX New topic

Span Property

Description	Sets or returns the Span time of either Gating or Time Domain transform windows
VB Syntax	<i>object.Span</i> = <i>value</i>
Variable	(Type) - Description
<i>object</i>	(object) As Gating or (object) As Transform
<i>value</i>	(double) - Span time in seconds. Choose any number between: 2*[(number of points-1) / frequency span] and 0
Return Type	Double
Default	20ns
Examples	<pre>Trans.Span = 4.5e-9 'sets the time span of a transform window - Write Gate.Span = 4.5e-9 'sets the Span time of a gating window -Write span = Trans.Span 'Read</pre>
C++ Syntax	HRESULT get_Span(double *pVal) HRESULT put_Span(double newVal)
Interface	ITransform IGating

Read-only

Span Property

Description	Returns the stimulus span of the measurement (stop-start data points). To understand how this property is useful, see IMeasurement2 Interface .
VB Syntax	<i>value</i> = <i>meas</i> . Span
Variable	(Type) - Description
	<i>value</i> (Double) - Variable to store the returned value.
	<i>meas</i> A Measurement (object)
Return Type	Double
Default	Not Applicable
Examples	<pre>Print meas.Span 'prints the span of the measurement</pre>
C++ Syntax	HRESULT get_Span(double * Val);
Interface	IMeasurement2

SParameterCalPorts Property

Description Read the final list of ports that will be fully calibrated by Calibrate All Channels. For the returned list of ports, specify connectors and cal kits for the calibration using the [GuidedCal](#) commands.

Ports to received ONLY a power cal (such as mixer LO ports) will NOT be on the returned list.

For each channel, specify the ports to be calibrated using [CalibrationPorts Property](#)

VB Syntax `ports = calAll.SParameterCalPorts`

Variable [\(Type\)](#) - [Description](#)

`ports` (Variant Array) Ports to be calibrated.

`calAll` A [CalibrateAllChannels](#) (object)

Return Type Variant Array

Default Not Applicable

Examples `ports = calAll.SParameterCalPorts 'returns the ports to be cal'd`

C++ Syntax HRESULT SParameterCalPorts([out,retval] VARIANT* calports);

Interface ICalibrateAllChannels

Last modified:

4-Jan-2012 New topic

SpectrumCenterFrequency Property

Description Sets and returns the receiver Center frequency for the IM Spectrum measurement. Valid ONLY when Tracking is NOT enabled and when [Sweep Type](#) = Linear. Otherwise, this setting is ignored.

VB Syntax *ims.SpectrumCenterFrequency = value*

Variable [\(Type\)](#) - Description

ims An [IMSpectrum](#) Object

value (Double) Center frequency in Hz. Choose a frequency within the range of the PNA.

Return Type Double

Default 1.0 GHz

Examples `ims.SpectrumCenterFrequency = 10e9 'Write`

`value = ims.SpectrumCenterFrequency 'Read`

C++ Syntax HRESULT get_SpectrumCenterFrequency(double *pVal)
HRESULT put_SpectrumCenterFrequency(double newVal)

Interface IIMSpectrum

Last Modified:

19-Aug-2008 MX New topic

SpectrumSpanFrequency Property

Description Sets and returns the Span of receiver frequencies for the IM Spectrum measurement. Valid ONLY when Tracking is NOT enabled and when [Sweep Type](#) = Linear. Otherwise, this setting is ignored.

VB Syntax *ims.SpectrumSpanFrequency = value*

Variable [\(Type\)](#) - Description

ims An [IMSpectrum](#) Object

value (Double) Frequency span in Hz. Choose a frequency within the range of the PNA.

Return Type Double

Default 100 MHz

Examples

```
ims.SpectrumSpanFrequency = 10e9 'Write
```

```
value = ims.SpectrumSpanFrequency 'Read
```

C++ Syntax HRESULT get_SpectrumSpanFrequency(double *pVal)
HRESULT put_SpectrumSpanFrequency(double newVal)

Interface IIMSpectrum

Last Modified:

19-Aug-2008 MX New topic

SpectrumStartFrequency Property

Description Sets and returns the receiver Start frequency for the IM Spectrum measurement. Valid ONLY when Tracking is NOT enabled and when Sweep Type = Linear. Otherwise, this setting is ignored.

VB Syntax *ims.SpectrumStartFrequency = value*

Variable [\(Type\)](#) - Description

ims An [IMSpectrum](#) Object

value (Double) Start frequency in Hz. Choose a frequency within the range of the PNA.

Return Type Double

Default 950 MHz

Examples `ims.SpectrumStartFrequency = 10e9 'Write`

`value = ims.SpectrumStartFrequency 'Read`

C++ Syntax HRESULT get_SpectrumStartFrequency(double *pVal)
HRESULT put_SpectrumStartFrequency(double newVal)

Interface IIMSpectrum

Last Modified:

19-Aug-2008 MX New topic

SpectrumStopFrequency Property

Description Sets and returns the receiver Stop frequency for the IM Spectrum measurement. Valid ONLY when Tracking is NOT enabled and when Sweep Type = Linear. Otherwise, this setting is ignored.

VB Syntax *ims.SpectrumStopFrequency = value*

Variable [\(Type\)](#) - Description

ims An [IMSpectrum](#) Object

value (Double) Stop frequency in Hz. Choose a frequency within the range of the PNA.

Return Type Double

Default 950 MHz

Examples `ims.SpectrumStopFrequency = 10e9 'Write`

`value = ims.SpectrumStopFrequency 'Read`

C++ Syntax HRESULT get_SpectrumStopFrequency(double *pVal)
HRESULT put_SpectrumStopFrequency(double newVal)

Interface IIMSpectrum

Last Modified:

19-Aug-2008 MX New topic

SSB_BalPortNegative Property

Description With a Single-ended - Single-ended - Balanced topology, returns the PNA port number that is connected to the Negative side of the DUT's Balanced Port.

Use [SetSSBPorts Method](#) to set the port mapping for a Single-Ended - Single-Ended - Balanced topology.

VB Syntax `var = balTopology.SSB_BalPortNegative`

Variable [\(Type\)](#) - Description

`var` (Long Integer) Variable to store the returned value.

`balTopology` A [BalancedTopology](#) (object)

Return Type Long Integer

Default Not Applicable

Examples `variable = balTopology.SSB_BalPortNegative` 'Read

C++ Syntax HRESULT get_SSB_BalPortNegative(long *bVal)

Interface IBalancedTopology

SSB_BalPortPositive Property

Description With a Single-ended - Single-ended - Balanced topology, returns the PNA port number that is connected to the Positive side of the DUT's Balanced Port.

Use [SetSSBPorts Method](#) to set the port mapping for a Single-Ended - Single-Ended - Balanced topology.

VB Syntax `var = balTopology.SSB_BalPortPositive`

Variable [\(Type\)](#) - Description

`var` (Long Integer) Variable to store the returned value.

`balTopology` A [BalancedTopology](#) (object)

Return Type Long Integer

Default Not Applicable

Examples `variable = balTopology.SSB_BalPortPositive` **'Read**

C++ Syntax HRESULT get_SSB_BalPortPositive(long *bVal)

Interface IBalancedTopology

SSB_SEPort1 Property

Description With a Single-ended - Single-ended - Balanced topology, returns the PNA port number that is connected to the DUT's Logical Port 1.

Use [SetSSBPorts Method](#) to set the port mapping for a Single-Ended - Single-Ended - Balanced topology.

VB Syntax *var* = *balTopology*.SSB_SEPort1

Variable [\(Type\)](#) - Description

var (Long Integer) Variable to store the returned value.

balTopology A [BalancedTopology](#) (object)

Return Type Long Integer

Default Not Applicable

Examples `variable = balTopology.SSB_SEPort1 'Read`

C++ Syntax HRESULT get_SSB_SEPort1(long *bVal)

Interface IBalancedTopology

SSB_SEPort2 Property

Description With a Single-ended - Single-ended - Balanced topology, returns the PNA port number that is connected to the DUT's Logical Port 2.

Use [SetSSBPorts Method](#) to set the port mapping for a Single-Ended - Single-Ended - Balanced topology.

VB Syntax `var = balTopology.SSB_SEPort2`

Variable [\(Type\)](#) - Description

`var` (Long Integer) Variable to store the returned value.

`balTopology` A [BalancedTopology](#) (object)

Return Type Long Integer

Default Not Applicable

Examples `variable = balTopology.SSB_SEPort2` 'Read

C++ Syntax HRESULT get_SSB_SEPort2(long *bVal)

Interface IBalancedTopology

SSBMeasurement Property

Description Sets and returns the measurement for the Single-Ended - Single-Ended - Balanced topology.

VB Syntax *balMeas.SSBMeasurement = value*

Variable [\(Type\)](#) - Description

balMeas A [BalancedMeasurement](#) (**object**)

value (**String**) - Single-ended - Single-ended - Balanced Measurement parameter. Not case sensitive. Choose from:

Sss11	Sss12	Ssd13	Ssc13
Sss21	Sss22	Ssd23	Ssc23
Sds31	Sds32	Sdd33	Sdc33
Scs31	Scs32	Scd33	Sc33
Imb1	Imb2	CMRR1	CMRR2
		(Sds31/Scs31)	(Sds32/Scs32)

Return Type String

Default Sss11

Examples `balMeas.SSBMeasurement = "Sss11" 'Write`

`variable = balMeas.SSBMeasurement 'Read`

C++ Syntax HRESULT get_SSBMeasurement(BSTR *pVal)

HRESULT put_SSBMeasurement(BSTR p newVal)

Interface IBalancedMeasurement

Stage1Coefficients Property

Description Sets and returns the digital filter coefficients of stage1.

VB Syntax *spm4.Stage1Coefficients = value*

Variable [\(Type\)](#) - Description

spm4 A [SignalProcessingModuleFour](#) (object)

value (**Variant Array**) Coefficients. An array of real values.

Return Type Variant

Default Stage dependent.

Examples `spm4.Stage1Coefficients = 0,0.1,0.7,0.7,0.1 'Write`
`mode = spm4.Stage1Coefficients 'Read`

C++ Syntax HRESULT get_Stage1Coefficients(VARIANT* pCoefs);
HRESULT put_Stage1Coefficients(VARIANT pCoefs);

Interface ISignalProcessingModuleFour

Last Modified:

18-Jun-2007 MX New topic

Stage1Frequency Property

Description Sets and returns the Numerically Controlled Oscillator (NCO) frequency of the Stage 1 filter. This command is only used when [FilterMode Property](#) is set to Manual.

VB Syntax *spm4.Stage1Frequency = value*

Variable [\(Type\) - Description](#)

spm4 A [SignalProcessingModuleFour](#) (**object**)

value (**Double**) Stage 1 Frequency. Min value= 0 Hz

Stage 1 Frequency. Min value= 0 Hz

With DSP 4 versions, Max value= 15 MHz.

With DSP 5 versions, Max value = 38 MHz.

[Learn more about DSP versions.](#)

Or programmatically use [MinimumIFFrequency Property](#) and [MaximumIFFrequency Property](#) to determine the range of settable values.

Return Type Double

Default Nominal IF Frequency. [Learn more](#)

Examples

```
spm4.Stage1Frequency = 9E6 'Write
mode = spm4.Stage1Frequency 'Read
```

C++ Syntax HRESULT get_Stage1Frequency(double *val);
 HRESULT put_Stage1Frequency(double val);

Interface ISignalProcessingModuleFour

Last Modified:

26-Aug-2010 Updated for DSP 5
 18-Jun-2007 MX New topic

Stage1MaximumCoefficient Property

Description Returns the maximum value of stage 1 coefficients.

VB Syntax *value* = *spm4*.Stage1MaximumCoefficient

Variable [\(Type\)](#) - Description

value **(Long)** Variable to store the returned Max coefficient.

spm4 A [SignalProcessingModuleFour](#) **(object)**

Default Not Applicable

Examples `mode = spm4.Stage1MaximumCoefficient 'Read`

C++ Syntax HRESULT get_Stage1MaximumCoefficient(long* val);

Interface ISignalProcessingModuleFour

Last Modified:

1-Jan-2007 MX New topic

Stage1MaximumCoefficientCount Property

Description Returns the maximum number of coefficients for Stage1.

VB Syntax *value* = *spm4*.Stage1MaximumCoefficientCount

Variable [\(Type\)](#) - Description

value **(Long)** Variable to store the returned Max coefficient count.

spm4 A [SignalProcessingModuleFour](#) **(object)**

Default Not Applicable

Examples `mode = spm4.Stage1MaximumCoefficientCount 'Read`

C++ Syntax HRESULT get_Stage1MaximumCoefficientCount(long* val);

Interface ISignalProcessingModuleFour

Last Modified:

1-Jan-2007 MX New topic

Stage1MaximumCoefficientSum Property

Description	Returns the maximum sum of all Stage1 coefficients.
VB Syntax	<i>value</i> = <i>spm4</i> . Stage1MaximumCoefficientSum
Variable	(Type) - Description
<i>value</i>	(__int64* val) Variable to store the returned Max sum of all coefficients.
<i>spm4</i>	A SignalProcessingModuleFour (object)
Default	Not Applicable
Examples	<code>mode = spm4.Stage1MaximumCoefficientSum 'Read</code>
C++ Syntax	HRESULT get_Stage1MaximumCoefficientSum(__int64* val);
Interface	ISignalProcessingModuleFour

Last Modified:

1-Jan-2007 MX New topic

Stage1MinimumCoefficientCount Property

Description Returns the minimum number of coefficients for Stage1

VB Syntax *value* = *spm4*.Stage1MinimumCoefficientCount

Variable [\(Type\)](#) - Description

value **(Long)** Variable to store the returned Min coefficient count.

spm4 A [SignalProcessingModuleFour](#) **(object)**

Default Not Applicable

Examples `mode = spm4.Stage1MinimumCoefficientCount 'Read`

C++ Syntax HRESULT get_Stage1MinimumCoefficientCount(long* val);

Interface ISignalProcessingModuleFour

Last Modified:

1-Jan-2007 MX New topic

Stage2Coefficients Property

Description Sets and returns Stage2Coefficients.

Note: Stage2 settings are ignored when using DSP Version 5. [Learn more.](#)

VB Syntax *spm4.Stage2Coefficients = value*

Variable [\(Type\)](#) - Description

spm4 A [SignalProcessingModuleFour](#) (**object**)

value (**VARIANT**) An array of real numbers. Filter coefficients

Return Type Variant

Default Not Applicable

Examples

```
spm4.Stage2Coefficients = 'Write
```

```
mode = spm4.Stage2Coefficients 'Read
```

C++ Syntax HRESULT get_Stage2Coefficients(VARIANT* pCoefs);

HRESULT put_Stage2Coefficients(VARIANT pCoefs);

Interface ISignalProcessingModuleFour

Last Modified:

1-Jan-2007 MX New topic

Stage2MaximumCoefficient Property

Description Returns the maximum value of stage 2 coefficients.

VB Syntax *value* = *spm4*.Stage2MaximumCoefficient

Variable [\(Type\)](#) - Description

value **(Long)** Variable to store the returned Max coefficient.

spm4 A [SignalProcessingModuleFour](#) **(object)**

Default Not Applicable

Examples `mode = spm4.Stage2MaximumCoefficient 'Read`

C++ Syntax HRESULT get_Stage2MaximumCoefficient(long* val);

Interface ISignalProcessingModuleFour

Last Modified:

1-Jan-2007 MX New topic

Stage2MaximumCoefficientCount Property

Description Returns the maximum number of coefficients for Stage2.

Note: Stage2 settings are ignored when using DSP Version 5.

[Learn more.](#)

VB Syntax `value = spm4.Stage2MaximumCoefficientCount`

Variable [\(Type\)](#) - Description

value **(Long)** Variable to store the returned Max coefficient count.

spm4 A [SignalProcessingModuleFour](#) (**object**)

Default Not Applicable

Examples `mode = spm4.Stage2MaximumCoefficientCount 'Read`

C++ Syntax HRESULT get_Stage2MaximumCoefficientCount(long* val);

Interface ISignalProcessingModuleFour

Last Modified:

1-Jan-2007 MX New topic

Stage2MaximumCoefficientSum Property

Description Returns the maximum sum of all Stage2 coefficients.

Note: Stage2 settings are ignored when using DSP Version 5.

[Learn more.](#)

VB Syntax `value = spm4.Stage2MaximumCoefficientSum`

Variable [\(Type\) - Description](#)

value (**__int64* val**) Variable to store the returned Max sum of all coefficients.

spm4 A [SignalProcessingModuleFour](#) (**object**)

Default Not Applicable

Examples `mode = spm4.Stage2MaximumCoefficientSum 'Read`

C++ Syntax HRESULT get_Stage2MaximumCoefficientSum(__int64* val);

Interface ISignalProcessingModuleFour

Last Modified:

1-Jan-2007 MX New topic

Stage2MinimumCoefficientCount Property

Description Returns the minimum number of coefficients for Stage2.

Note: Stage2 settings are ignored when using DSP Version 5.

[Learn more.](#)

VB Syntax `value = spm4.Stage2MinimumCoefficientCount`

Variable [\(Type\)](#) - Description

value **(Long)** Variable to store the returned Min coefficient count.

spm4 A [SignalProcessingModuleFour](#) (**object**)

Default Not Applicable

Examples `mode = spm4.Stage2MinimumCoefficientCount 'Read`

C++ Syntax HRESULT get_Stage2MinimumCoefficientCount(long* val);

Interface ISignalProcessingModuleFour

Last Modified:

1-Jan-2007 MX New topic

Stage3FilterType Property

Description Sets and returns the Stage 3 filter type. This command is only used when [FilterMode](#) is set to Manual.

VB Syntax `spm4.Stage3FilterType = value`

Variable [\(Type\)](#) - Description

spm4 A [SignalProcessingModuleFour](#) (object)

value (String) Filter type. Chose from:

- "RECT" Rectangular Window Filter
- "TUKEY" Tukey Filter
- "PWIN" Pulse window filter

Default TUKEY

Examples

```
spm4.Stage3FilterType = "PWIN"  
mode = spm4.Stage3FilterType 'Read'
```

C++ Syntax HRESULT get_Stage3FilterType(BSTR* pFType);
HRESULT put_Stage3FilterType(BSTR GType);

Interface ISignalProcessingModuleFour

Last Modified:

1-Jan-2007 MX New topic

Stage3FilterTypes Property

Description	Returns a list of strings for the currently supported filter types that can be used for the stage 3 filter. This command is only used when FilterMode is set to False (Manual). See Stage3FilterType for a list of currently supported filter types.
VB Syntax	<i>values</i> = <i>spm4</i> .Stage3FilterTypes
Variable	(Type) - Description
<i>value</i>	(Variant) Variable to store the returned filter types.
<i>spm4</i>	A SignalProcessingModuleFour (object)
Return Type	Variant Array
Default	Not Applicable
Examples	<pre>mode = spm4.Stage3FilterTypes 'Read</pre>
C++ Syntax	HRESULT get_Stage3FilterTypes(VARIANT* pTypes);
Interface	ISignalProcessingModuleFour

Last Modified:

1-Jan-2007 MX New topic

Stage3Parameter Property

Description Sets and returns the Stage 3 filter parameters.

Must first select the filter type using [Stage3FilterType](#) before setting these parameters

Use [Stage3Parameters](#) to return a list of the available parameters for the currently selected filter type.

VB Syntax `spm4.Stage3Parameter(param) = value`

Variable [\(Type\)](#) - Description

spm4 A [SignalProcessingModuleFour](#) (**object**)

param (**String**) Filter parameter. Choose from:

"C" - Tap count (Tukey, RECT, PWIN)

"P" - Period (PWIN ONLY)

"D" - Delay (PWIN ONLY)

"W" - Width (PWIN ONLY)

"R" - Ramp Count (PWIN ONLY)

"M" - Number of times to repeat the user-supplied array for each data point (COEF ONLY)

value (**String**) Parameter Value for the specified stage 3 parameter. Use [Stage3ParameterMaximum](#) and [Stage3ParameterMinimum](#) to return a range of values for the specified parameter.

Default **RECT:** C = 1

PWIN: C=1E6, P=10ms, D=50us, W=50us, R=7

TUKEY: C=1

Examples

```
spm4.Stage3Parameter("C") = 2
```

```
mode = spm4.Stage3Parameter("pwin") 'Read
```

C++ Syntax HRESULT get_Stage3Parameter(BSTR pName, double* pVal);

HRESULT put_Stage3Parameter(BSTR pName, double pVal);

Interface ISignalProcessingModuleFour

Last Modified:

1-Jan-2007 MX New topic

Stage3ParameterMaximum Property

Description	Returns maximum parameter value for the current filter type.
VB Syntax	<i>value</i> = <i>spm4</i> . Stage3ParameterMaximum (<i>parameter</i>)
Variable	(Type) - Description
<i>value</i>	(Variant) Variable to store the maximum parameter value.
<i>spm4</i>	A SignalProcessingModuleFour (object)
<i>parameter</i>	(String) Parameter name. See Stage3Parameter Property for a list of parameters.
Return Type	Double
Default	Not Applicable
Examples	<pre>mode = spm4.Stage3ParameterMaximum ("c") 'Read</pre>
C++ Syntax	HRESULT get_Stage3ParameterMaximum(BSTR pName, double* pVal);
Interface	ISignalProcessingModuleFour

Last Modified:

1-Jan-2007 MX New topic

Stage3ParameterMinimum Property

Description	Returns minimum parameter value for the current filter type.
VB Syntax	<i>value</i> = <i>spm4</i> . Stage3ParameterMinimum (<i>parameter</i>)
Variable	(Type) - Description
<i>value</i>	(Variant) Variable to store the minimum parameter value.
<i>spm4</i>	A SignalProcessingModuleFour (object)
<i>parameter</i>	(String) Parameter name. See Stage3Parameter Property for a list of parameters.
Return Type	Double
Default	Not applicable
Examples	<pre>mode = spm4.Stage3ParameterMinimum ("c") 'Read</pre>
C++ Syntax	HRESULT get_Stage3ParameterMinimum(BSTR pName, double* pVal);
Interface	ISignalProcessingModuleFour

Last Modified:

1-Jan-2007 MX New topic

Stage3Parameters Property

Description	Returns the names of parameters for the current filter type. Use Stage3FilterType Property to set the filter type.
VB Syntax	<i>values</i> = <i>spm4</i> .Stage3Parameters
Variable	(Type) - Description
<i>value</i>	(Variant) Variable to store the returned parameter names.
<i>spm4</i>	A SignalProcessingModuleFour (object)
Return Type	Variant
Default	Not Applicable
Examples	<code>mode = spm4.Stage3Parameters 'Read</code>
C++ Syntax	HRESULT get_Stage3Parameters(VARIANT* pNames);
Interface	ISignalProcessingModuleFour

Last Modified:

18-Jan-2007 MX New topic

StandardDeviation Property

Description Returns the standard deviation of the measurement.
To retrieve all 3 statistics value at the same time, use [meas.GetTraceStatistics](#)

VB Syntax `stdev = meas.StandardDeviation`

Variable [\(Type\)](#) - **Description**

stdev **(single)** - Variable to store standard deviation value

meas A Measurement **(object)**

Return Type Single

Default Not applicable

Examples `stdev = meas.StandardDeviation 'Read`

C++ Syntax HRESULT get_StandardDeviation(float* stdDeviation)

Interface IMeasurement

StandardForClass Property - **Superseded**

Description **Superseded** This command sets a **single** standard to a calibration class. Does NOT set or dictate the order for measuring the standards.

Use [GetStandardForClass](#) and [SetStandardForClass](#). These commands allow up to seven standards to be assigned to a cal class.

VB Syntax *calKit.StandardForClass(class, portNum) = value*

Variable **(Type) - Description**

calKit A CalKit (**object**). Use calKit.[GetCalStandard](#) to get a handle to the standard.

class (**enum NACalClass**) Standard. Choose from:

- 1 - naClassA
- 2 - naClassB
- 3 - naClassC
- 4 - naClassD
- 5 - naClassE
- 6 - naReferenceRatioLine
- 7 - naReferenceRatioThru

SOLT Standards

- 1 - naSOLT_Open
- 2 - naSOLT_Short
- 3 - naSOLT_Load
- 4 - naSOLT_Thru
- 5 - naSOLT_Isolation

TRL Standards

- 1 - naTRL_Reflection
- 2 - naTRL_Line_Reflection
- 3 - naTRL_Line_Tracking
- 4 - naTRL_Thru
- 5 - naTRL_Isolation

portNum (**long**) - The port number the standard will be connected to. For example, you may have a 3.5mm connector designated for port 1, and Type N designated for port 2.

value (**long**) - Calibration class number. Choose a number between **1** and **8**. The *<value>* numbers are associated with the following calibration classes:

<i><value></i>	Class	Description
1	S11A	Reflection standard
2	S11B	Reflection standard
3	S11C	Reflection standard
4	S21T	Thru standard
5	S22A	Reflection standard
6	S22B	Reflection standard
7	S22C	Reflection standard
8	S21T	Thru standard

Return Type Long Integer

Default Not Applicable

Examples
`calKit.StandardForClass(naSOLT_Short, 1) = 1`
`Kclass = calKit.StandardForClass(naSOLT_Short, 1)`

C++ Syntax
HRESULT put_StandardForClass (NACalClass item, long pNum);
HRESULT get_StandardForClass (NACalClass* item, long *pNum);

Interface ICalKit

StartFrequency (Cal Set) Property

Description Returns the start frequency that is stored in the Cal Set.

VB Syntax *value* = *CalSet*.**StartFrequency** (*range*)

Variable [\(Type\)](#) - Description

value **(double)** - returned Start frequency in Hertz.

CalSet [CalSet](#) **(object)**

range **(Long)** Choose: **0** (Source and receiver frequency)

Return Type Double

Default Not Applicable

Examples

```
start = calset.StartFrequency(0) 'Reads the start frequency
stored in the cal set.
```

C++ Syntax HRESULT get_StartFrequency(long range, *pVal)

Interface |CalSet3

Last modified:

25-Jan-2011 Fixed range argument

Nov. 1, 2006 New command - split from ch.StartFreq

StartFrequency Property

Description Sets or returns the start frequency of the channel. (Channel Object)

Sets or returns the start frequency of the segment. (Segment Object)

Sets or returns the start frequency of the FOM Range. (FOMRange Object)

Sets or returns the start frequency of the Power Sensor coverage (GuidedCalibrationPowerSensor Object)

See also [Measurement2](#) interface

VB Syntax *object.StartFrequency = value*

Variable [\(Type\)](#) - Description

object Any of the following:

[Channel](#) (**object**)

[Segment](#) (**object**)

[FOMRange](#) (**object**)

[GuidedCalibrationPowerSensor](#) (**object**)

value (**double**) - Start frequency in Hertz. Choose any number between the **minimum** and **maximum** frequencies of the analyzer.

Return Type Double

Default Channel - Minimum frequency of the analyzer

Segment - 0

FOMRange - Minimum frequency of the analyzer

PowerSensor - Minimum frequency of the analyzer

Examples

```
chan.StartFrequency = 4.5e9 'sets the start frequency of a linear sweep
for the channel object -Write
```

```
startfreq = Chan.StartFrequency 'Read
```

C++ Syntax HRESULT get_StartFrequency(double *pVal)

HRESULT put_StartFrequency(double newVal)

Interface IChannel
ISegment
IFOMRange
IGuidedCalibrationPowerSensor

Last modified:

8-Feb-2011 Added Power Sensor
8-Mar-2007 Added FOMRange
1-Nov-2006 Removed Cal Set object. There is now a new [cs.StartFreq](#)

StartPower Property

Description Sets the start power of the analyzer when [sweep type](#) is set to Power Sweep. Frequency of the measurement is set with chan.[CWFrequency](#).

VB Syntax *object*.**StartPower** = *value*

Variable [\(Type\)](#) - Description

object One of the following:

- [Channel](#) (object)
- [CalSet](#) (object) - Read-only property

value **(double)** - Start Power in dBm.

Note: The range of settable power values depends on the PNA model and if source attenuators are installed. To determine the range of values, use [cap.MaximumSourceALCPower](#) and [cap.MinimumSourceALCPower](#)

Auto attenuation is not allowed in Power Sweep.

Return Type Double

Default 0

Examples

```
Chan.StartPower = -10 'Write
strtpwr = Chan.StartPower 'Read
```

C++ Syntax HRESULT get_StartPower(double *pVal)
HRESULT put_StartPower(double newVal)

Interface IChannel

|CalSet3

Last Modified:

7-Jan-2008 Removed FOMRange
8-Mar-2007 Added FOMRange

Start Property

Description	Sets or returns the start time of either Gating or Time Domain transform windows
VB Syntax	<i>object</i> .Start = <i>value</i>
Variable	(Type) - Description
<i>object</i>	(object) As Gating or (object) As Transform
<i>value</i>	(double) - Start time in seconds. Choose any number between: ± (number of points-1) / frequency span
Return Type	Double
Default	-10ns
Examples	<pre>Trans.Start = 4.5e-9 'sets the start time of a transform window -Write Gate.Start = 4.5e-9 'sets the start time of a gating window - Write strt = Trans.Start 'Read</pre>
C++ Syntax	HRESULT get_Start(double *pVal) HRESULT put_Start(double newVal)
Interface	ITransform IGating

Read-only

Start Property

Description	Returns the stimulus value of the first data point for the measurement. To understand how this property is useful, see IMeasurement2 Interface .
VB Syntax	<i>value</i> = <i>meas</i> . Start
Variable	(Type) - Description
<i>value</i>	(Double) - Variable to store the returned value
<i>meas</i>	A Measurement (object)
Return Type	Double
Default	Not Applicable
Examples	<pre>Print meas.Start 'prints the stimulus value of the first data point</pre>
C++ Syntax	HRESULT get_Start (double * Val);
Interface	IMeasurement2

Start Property

Description Sets and returns start DC value for the specified DC source.

VB Syntax *dc.Start(name,port) = value*

Variable [\(Type\)](#) - Description

dc An [DCStimulus](#) (object)

name,port (String) Name of the "DC source, port"

Use [Source Property](#) to read a list of configured DC source names.

To set the DC source to be always ON, do NOT specify a port.

value (Double) DC Start value. Choose a value within the range of the DC source.

Return Type Double

Default .5

Examples

```
'Set A01 to always ON
dc.Start "A01", 3

'Read Start for MyDCSource,Port 1
dc.Start? "MyDCSource,Port 1"
```

C++ Syntax HRESULT get_Start(BSTR name, VARIANT_BOOL * pValue);
HRESULT put_Start(BSTR name, VARIANT_BOOL newValue);

Interface IDCSstimulus

Last Modified:

21-Feb-2012 MX New topic

StartFrequency Property

Description	Sets and returns the phase reference cal start frequency.
VB Syntax	<i>phasRef.StartFrequency = value</i>
Variable	(Type) - Description
<i>phasRef</i>	A PhaseReferenceCalibration Object
<i>value</i>	Start frequency in Hz. Choose any frequency from 17.5 MHz to the stop frequency of the PNA.
Return Type	Double
Default	17.5e6
Examples	<pre>phase.StartFrequency = 20e6</pre> See example program
C++ Syntax	HRESULT get_StartFrequency(Double* pVals); HRESULT put_StartFrequency(Double pVals);
Interface	IPhaseReferenceCalibration

Last Modified:

3-Apr-2012 MX New topic

StartPhase Property

Description	Write and read the start value of phase sweep. Must also send Sweep Type Property to put the analyzer into phase sweep mode.
VB Syntax	<i>phase</i> .StartPhase(<i>srcPort</i>) = <i>value</i>
Variable	(Type) - Description
<i>phase</i>	A PhaseControl Object
<i>srcPort</i>	(Long Integer) Source port for which to make phase control settings.
	Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use chan.getPortNumber to translate the string into a port number. To learn more see Remotely Specifying a Source Port
<i>value</i>	(Double) Start phase value in degrees. Choose a value between -360 and 360.
Return Type	Double
Default	0 degrees
Examples	<pre>phase.StartPhase 1 = 60 ' Write value = phase.StartPhase 2 ' Read</pre>
C++ Syntax	HRESULT get_StartPhase(long port, double* pVal); HRESULT put_StartPhase(long port, double newVal);
Interface	IPhaseControl

Last Modified:

8-Dec-2010 MX New topic

StartPowerEx Property

Description Sets and reads the power sweep start power value for a specific port. This allows uncoupled forward and reverse power sweep ranges. Must also set [SweepType](#) = `naPowerSweep`, [Coupled](#) = `False` (Off), and [StopPowerEx](#).

VB Syntax `chan.StartPowerEx (srcPort) = value`

Variable [\(Type\)](#) - Description

`chan` A [Channel](#) (object)

`srcPort` (**long integer**) – Source port for which to set the Start power value.

Note: If the port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-x model, then you must use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#).

`value` (**double**) - Start Power in dBm.

Note: The range of settable power values depends on the PNA model and if source attenuators are installed. To determine the range of values, use [cap.MaximumSourceALCPower](#) and [cap.MinimumSourceALCPower](#)

Auto attenuation is not allowed in Power Sweep.

Return Type Double

Default -10 dBm

Examples `Chan.StartPowerEx 1 = -10 'Write`

`strtpwr = Chan.StartPowerEx 2 'Read`

C++ Syntax `HRESULT get_StartPowerEx(long port, double *pVal)`
`HRESULT put_StartPowerEx(long port, double newVal)`

Interface `IChannel13`

Last Modified:

23-May-2008 MX New topic

StartRatioedPower Property

Description Write and read the start power ratioed value. Must also set [SweepType](#) to Power.

VB Syntax `phase.StartRatioedPower(srcPort) = value`

Variable [\(Type\)](#) - Description

phase A [PhaseControl](#) Object

srcPort (Long Integer) Source port for which to make phase control settings.

Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#)

value (Double) Start power ratio value in dBc. Must be within the allowable range of the PNA.

Return Type Double

Default 0 dBc

Examples

```
phase.StartRatioedPower 1 = -1 ' Write
```

```
value = phase.StartRatioedPower 2 ' Read
```

C++ Syntax

```
HRESULT get_StartRatioedPower(long port, double* pVal);
```

```
HRESULT put_StartRatioedPower(long port, double newVal);
```

Interface IPhaseControl

Last Modified:

8-Dec-2010 MX New topic

State (GPL) Property

Description Enables and disables Global Power Limiting for the specified port.

VB Syntax `gpl.State(port) = bool`

Variable [\(Type\)](#) - Description

gpl A [GlobalPowerLimit](#) (**object**)

port (**Long**) Port number for which power limit state is to be set.

bool (**Boolean**) Choose from:

True - Power Limiting is enabled.

False - Power Limiting is disabled.

Return Type Boolean

Default False

Examples `gpl.State(1) = True 'Write`

`Limit = gpl.State(2) 'Read`

C++ Syntax HRESULT get_State(long port, VARIANT_BOOL *pVal)
HRESULT put_Limit(long port, VARIANT_BOOL newVal)

Interface IGlobalPowerLimit

Last Modified:

10-Aug-2009 MX New topic

State Property

Description Turns an Object ON and OFF.

VB Syntax *object.State* = *value*

Variable [\(Type\)](#) - Description

object Applies to any of the following objects:

[FOM](#)

[Gating](#)

[InterfaceControl](#)

[LimitTest](#)

Port Extension- Superseded. See either:

- [FixturingState Property](#)
- [PortExtState Property](#)

[Segment](#)

[Transform](#)

[Equation](#)

[FIFO](#)

Notes:

- **LimitTest.State** - If using Global Pass/Fail status, trigger the PNA AFTER turning Limit testing ON.
- **Segment.State** - At least ONE segment must be ON or [Sweep Type](#) is automatically set to **Linear**.

value (boolean) -

False - Turns *obj* OFF

True - Turns *obj* ON

Return Type Boolean

Default Depends on the object:

0 - FOM

0 - Gating

0 - InterfaceControl

0 - LimitTest

1 - Segment

0 - Transform

0 - Equation

0 - FIFO

Examples

```
Seg.State = 1 'Turns the segment object ON -Write
```

```
tran = Trans.State 'returns the state of Transform -Read
```

C++ Syntax

```
HRESULT get_State(VARIANT_BOOL *pVal)  
HRESULT put_State(VARIANT_BOOL newVal)
```

Interface

ISegment

IInterfaceControl

ITransform

IGating

ILimitTest

IFOM

IEquation

IEmbeddedLO

IFIFO

Last Modified:

5-Nov-2012 Updated links to states

10-Oct-2008 Added FIFO

13-Apr-2007 Added EmbeddedLO

State Property

Description Sets and returns the ON / Off state of the specified DC source and port.

VB Syntax *dc.State(name,port) = state*

Variable [\(Type\)](#) - Description

dc An [DCStimulus](#) (object)

name,port (String) Name of the "DC source, port"

Use [Source Property](#) to read a list of configured DC source names.

To set the DC source to be always ON, do NOT specify a port.

state Boolean. ON / Off state. Choose from:

True - DC source/port enabled.

False - DC source/port disabled.

Return Type Boolean

Default False

Examples

```
'Set A01 to always ON
dc.State "A01",True

'Set MyDCSource to ON when the RF source for Port 1
is ON
dc.State "MyDCSource,Port 1",ON

'Read state for MyDCSource,Port 1
dc.State? "MyDCSource,Port 1"
```

C++ Syntax HRESULT get_State(BSTR name, VARIANT_BOOL * pValue);

HRESULT put_State(BSTR name, VARIANT_BOOL
newValue);

Interface IDCStimulus

21-Feb-2012 MX New topic

State (Rx Leveling) Property

Description	Sets and reads the state of Receiver Leveling for a specific source port.
VB Syntax	<i>RxLevel.State(srcPort) = value</i>
Variable	(Type) - Description
<i>RxLevel</i>	A ReceiverLeveling Object
<i>value</i>	(Boolean) Choose from: True - Receiver leveling ON False - Receiver leveling OFF
<i>srcPort</i>	(Long Integer) Source port for which to set the state of Receiver Leveling. Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use chan.getPortNumber to translate the string into a port number. To learn more see Remotely Specifying a Source Port
Return Type	Variant Boolean
Default	False
Examples	<pre>rxLevel.State (1) = True ' Write value = rxLevel.State 2' Read</pre>
C++ Syntax	<pre>HRESULT get_State(long port, VARIANT_BOOL* pLevelingState); HRESULT put_State(long port, VARIANT_BOOL LevelingState);</pre>
Interface	IRceiverLevelingConfiguration

Last Modified:

12-Feb-2009 MX New topic

State Property

Description Turns the specified pulse generator ON and OFF.

VB Syntax *pulse.State* (*n*) = *value*

Variable [\(Type\)](#) - Description

pulse A [PulseGenerator](#) (object)

n (**Integer**) Pulse generator number. Choose from 0 to 4.

Or use [PulseGeneratorID](#) to refer to an external pulse generator.

0 is the generator that pulses the ADC.

value **True** - turns pulse output ON.

False - turns pulse output OFF.

Return Type Boolean

Default False

Examples `pulse.State(1) = True 'Write`

`value = pulse.State(4) 'Read`

C++ Syntax HRESULT get_State(VARIANT_BOOL *pVal);

HRESULT put_State(VARIANT_BOOL newVal);

Interface IPulseGenerator

Last Modified:

16-Feb-2012 Edit for Ext. Pulse Gens

2-Jan-2007 MX New topic

StatisticsRange Property

Description Sets the User Range number for calculating measurement statistics. Set the start and stop values for a User Range with [UserRangeMin](#) and [UserRangeMax](#).

There are 16 User Ranges per channel. User ranges are applied independently to any measurement.

VB Syntax *meas.StatisticsRange* = *value*

Variable [\(Type\)](#) - Description

meas A Measurement (**object**)

value (**long integer**) - Range Number. Choose any number between 0 and 16

0 is Full Span

1 - 16 are user-defined ranges

Return Type Long Integer

Default 0

Examples

```
meas.StatisticsRange = 2 'Write
```

```
stairange = meas.StatisticsRange 'Read
```

C++ Syntax HRESULT get_StatisticsRange(long* rangeNumber)
HRESULT put_StatisticsRange(long rangeNumber)

Interface IMeasurement

StatusAsString Property

Description Returns a string that describes the result of the last tuning sweeps.

VB Syntax *value* = *embedLO*.**StatusAsString**

Variable [\(Type\)](#) - Description

value **(String)** Variable to store the returned data.

embedLO An [EmbeddedLODiagnostic](#) **(object)**

Return Type **(String)**

Default Not Applicable

Examples `data= embedLO.StatusAsString 'read'`

C++ Syntax HRESULT get_StatusAsString(BSTR* status);

Interface IEmbeddedLODiagnostic

Last Modified:

12-Apr-2007 MX New topic

StepRiseTime Property

Description	Sets or returns the Rise time of the stimulus in Low Pass Step Mode.
VB Syntax	<i>trans</i> .StepRiseTime = <i>value</i>
Variable	(Type) - Description
<i>trans</i>	A Transform (object)
<i>value</i>	(double) - Rise time in seconds. Choose any number between 5.0e-13 and 1.63e-14 .
Return Type	Double
Default	0
Examples	<pre>trans.StepRiseTime = 1.0e-14 'sets the step rise time to 100 psec. -Write</pre> <pre>rt = trans.StepRiseTime 'Read</pre>
C++ Syntax	HRESULT get_StepRiseTime(double *pVal) HRESULT put_StepRiseTime(double newVal)
Interface	ITransform

StepData Property

Description Returns an array of data from the specified tuning sweep.

VB Syntax *value* = *embedLO*.StepData (*n*)

Variable [\(Type\)](#) - Description

value (**Variant Array**) Variable to store the returned data.

embedLO An [EmbeddedLODiagnostic](#) (**object**)

n (Long) Tuning sweep number. Use [NumberOfSweeps](#) to find the number of sweeps taken.

Default Not Applicable

Examples `data= embedLO.StepData 3 'read`

C++ Syntax HRESULT StepData(long sweep,VARIANT* pArray);

Interface IEmbeddedLODiagnostic

Last Modified:

12-Apr-2007 MX New topic

StepTitle Property

Description Returns the title of the specified tuning sweep.

VB Syntax *value* = *embedLODiag*.**StepTitle** (*n*)

Variable [\(Type\)](#) - Description

value **(String)** Variable to store the returned data.

embedLODiag An [EmbeddedLODiagnostic](#) (**object**)

n (Long) Tuning sweep number. Use [NumberOfSweeps](#) to find the number of sweeps taken.

Default Not Applicable

Examples `data= embedLO.StepTitle 3 'read'`

C++ Syntax HRESULT StepTitle (long sweep, BSTR * title);

Interface IEmbeddedLODiagnostic

Last Modified:

13-Apr-2007 MX New topic

StimulusValues Property

Description	Returns the specified X-axis FOM frequency range. The array contains one frequency value for each data point.
VB Syntax	<i>value</i> = <i>calSet</i> . StimulusValues (<i>range</i>)
Variable	(Type) - Description
<i>calSet</i>	CalSet (object) - Read-only property
<i>range</i>	(Long) FOM frequency range to read. 0 - returns source frequencies. 1 - returns response frequencies. 2 - returns primary frequencies.
Return Type	1-dimensional variant array
Default	Not Applicable
Examples	<pre>array = CalSet.StimulusValues 'Read</pre>
C++ Syntax	HRESULT get_StimulusValues (long range, VARIANT* vals)
Interface	ICalSet3

Last Modified:

16-Sep-2009 Added 2 - primary
19-Oct-2007 Fixed title and syntax

Stop Property

Description Sets and returns Stop DC value for the specified DC source.

VB Syntax *dc.Stop (name,port) = value*

Variable [\(Type\)](#) - Description

dc An [DCStimulus](#) (object)

name,port (String) Name of the "DC source, port"

Use [Source Property](#) to read a list of configured DC source names.

To set the DC source to be always ON, do NOT specify a port.

value (Double) DC Stop value. Choose a value within the range of the DC source.

Return Type Double

Default 1

Examples

```
'Set AO1 to always ON
dc.Stop "AO1", 3

'Read Stop for MyDCSource,Port 1
dc.Stop? "MyDCSource,Port 1"
```

C++ Syntax HRESULT get_Stop(BSTR name, VARIANT_BOOL * pValue);
HRESULT put_Stop(BSTR name, VARIANT_BOOL newValue);

Interface IDCSstimulus

Last Modified:

21-Feb-2012 MX New topic

StopFrequency Property

Description Sets or returns the stop frequency of the channel. (Channel Object)

Sets or returns the stop frequency of the segment. (Segment Object)

Sets or returns the stop frequency of the FOM Range. (FOMRange Object)

Sets or returns the stop frequency of the Power Sensor coverage (GuidedCalibrationPowerSensor Object)

Sets or returns the stop frequency of the Phase Reference Calibration.

See also [Measurement2](#) interface.

VB Syntax *object*.StopFrequency = *value*

Variable [\(Type\)](#) - Description

object Any of the following:

[Channel](#) (object)

[Segment](#) (object)

[FOMRange](#) (object)

[GuidedCalibrationPowerSensor](#) (object)

[PhaseReferenceCalibration](#) (object)

value (**double**) - Stop frequency in Hertz. Choose any number between the **minimum** and **maximum** frequencies of the analyzer.

Return Type Double

Default Channel - Maximum frequency of the analyzer.

Segment - 0

FOMRange - Maximum frequency of the analyzer.

GuidedCalibrationPowerSensor - Maximum frequency of the analyzer.

PhaseReferenceCalibration - Maximum frequency of the analyzer.

Examples

```
chan.StopFrequency = 4.5e9 'sets the stop frequency for the channel  
object -Write
```

```
stopfreq = Chan.StopFrequency 'Read
```

C++ Syntax

```
HRESULT get_StopFrequency(double *pVal)
```

```
HRESULT put_StopFrequency(double newVal)
```

Interface

```
IChannel
```

```
ISegment
```

```
IFOMRange
```

```
IGuidedCalibrationPowerSensor
```

Last modified:

- 3-Apr-2012 Added PhaseReferenceCalibration
- 8-Feb-2011 Added GuidedCalibrationPowerSensor
- 8-Mar-2007 Added FOMRange
- 1-Nov-2006 Removed Cal Set object - created cs.stopfreq

StopFrequency (Cal Set) Property

Description	Returns the stop frequency that is stored in the Cal Set.
VB Syntax	<i>value</i> = <i>CalSet</i> . StopFrequency (<i>range</i>)
Variable	(Type) - Description
<i>value</i>	(double) - returned Stop frequency in Hertz.
<i>CalSet</i>	CalSet (object)
<i>range</i>	(Long) Choose: 0 (Source and receiver frequency)
Return Type	Double
Default	Not Applicable
Examples	<pre>stop = calset.StopFrequency(0) 'Reads the stop frequency stored in the cal set.</pre>
C++ Syntax	HRESULT get_StopFrequency(long range, double *pVal)
Interface	CalSet3

Last modified:

25-Jan-2011 Fixed range argument
Nov. 1, 2006 New command - split from ch.StopFreq

StopPower Property

Description Sets the Stop Power of the analyzer when [sweep type](#) is set to Power Sweep. Frequency of the measurement is set with chan.[CWFrequency](#).

VB Syntax *object*.**StopPower** = *value*

Variable [\(Type\)](#) - [Description](#)

object One of the following:

- [Channel](#) (object)
- [CalSet](#) (object) - Read-only property

value **(double)** - Stop Power in dB.

Note: The range of settable power values depends on the PNA model and if source attenuators are installed. To determine the range of values, use [cap.MaximumSourceALCPower](#) and [cap.MinimumSourceALCPower](#)

Auto attenuation is not allowed in Power Sweep.

Return Type Double

Default 0

Examples

```
Chan.StopPower = -10 'Write
stppwr = Chan.StopPower 'Read
```

C++ Syntax HRESULT get_StopPower(double *pVal)
HRESULT put_StopPower(double newVal)

Interface IChannel
|CalSet3

Last Modified:

7-Jan-2008 Removed FOMRange
8-Mar-2007 Added FOMRange

Stop Property

Description Sets or returns the Stop time of either Gating or Time Domain transform windows

VB Syntax *object*.**Stop** = *value*

Variable [\(Type\)](#) - Description

object **(object)** As Gating

or

(object) As Transform

value **(double)** - Start time in seconds. Choose any number between:
 \pm **(number of points-1) / frequency span**

Return Type Double

Default 10 ns

Examples

```
Trans.Stop = 4.5e-9 'sets the stop time of a transform window -  
Write  
Gate.Stop = 4.5e-9 'sets the stop time of a gating window -Write  
stp = Trans.Stop 'Read
```

C++ Syntax HRESULT get_Stop(double *pVal)
HRESULT put_Stop(double newVal)

Interface ITransform
IGating

Read- only

Stop Property

Description Returns the stimulus value of the last data point for the measurement. To understand how this property is useful, see [IMeasurement2 Interface](#).

VB Syntax *value* = *meas*.**Stop**

Variable [\(Type\)](#) - Description

value **(Double)** Variable to store the returned value

meas A Measurement **(object)**

Return Type Double

Default Not Applicable

Examples `Print meas.Stop` 'prints the stimulus value of the last data point

C++ Syntax HRESULT get_Stop(double * Val);

Interface IMeasurement2

StopPhase Property

Description	Write and read the stop value of phase sweep. Must also send Sweep Type Property to put the analyzer into phase sweep mode.
VB Syntax	<i>phase</i> . StopPhase (<i>srcPort</i>) = <i>value</i>
Variable	(Type) - Description
<i>phase</i>	A PhaseControl Object
<i>srcPort</i>	(Long Integer) Source port for which to make phase control settings.
	Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use chan.getPortNumber to translate the string into a port number. To learn more see Remotely Specifying a Source Port
<i>value</i>	(Double) Stop phase value in degrees. Choose a value between -360 and 360.
Return Type	Double
Default	0 degrees
Examples	<pre>phase.StopPhase 1 = 60 ' Write value = phase.StopPhase 2 ' Read</pre>
C++ Syntax	HRESULT get_StopPhase(long port, double* pVal); HRESULT put_StopPhase(long port, double newVal);
Interface	IPhaseControl

Last Modified:

8-Dec-2010 MX New topic

StopPowerEx Property

Description Sets and reads the power sweep start power value for a specific port. This allows uncoupled forward and reverse power sweep ranges. Must also set [SweepType](#) = `naPowerSweep`, [Coupled](#) = `False` (Off), and [StartPowerEx](#).

VB Syntax `chan.StopPowerEx (srcPort) = value`

Variable [\(Type\)](#) - Description

`chan` A [Channel](#) (object)

`srcPort` (**long integer**) – Source port for which to set the Stop power value.

Note: If the port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-x model, then you must use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#).

`value` (**double**) - Stop Power in dBm.

Note: The range of settable power values depends on the PNA model and if source attenuators are installed. To determine the range of values, use [cap.MaximumSourceALCPower](#) and [cap.MinimumSourceALCPower](#)

Auto attenuation is not allowed in Power Sweep.

Return Type Double

Default 0 dBm

Examples

```
Chan.StopPowerEx 1 = -10 'Write
stopPwr = Chan.StopPowerEx 2 'Read
```

C++ Syntax `HRESULT get_StopPowerEx(long port, double *pVal)`
`HRESULT put_StopPowerEx(long port, double newVal)`

Interface `IChannel13`

StopRatioedPower Property

Description Write and read the stop power ratioed value. Must also set [SweepType](#) to Power.

VB Syntax *phase*.StopRatioedPower(*srcPort*) = *value*

Variable [\(Type\)](#) - Description

phase A [PhaseControl](#) Object

srcPort (Long Integer) Source port for which to make phase control settings.

Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#)

value (Double) Stop power ratio value in dBc. Must be within the allowable range of the PNA.

Return Type Double

Default 0 dBc

Examples

```
phase.StopRatioedPower 1 = -1 ' Write
```

```
value = phase.StopRatioedPower 2 ' Read
```

C++ Syntax

```
HRESULT get_StopRatioedPower(long port, double* pVal);
```

```
HRESULT put_StopRatioedPower(long port, double newVal);
```

Interface IPhaseControl

Last Modified:

8-Dec-2010 MX New topic

strPort2Pdeembed_S2PFile Property

Description Sets and returns the 2 port De-embedding .S2P file name for the specified port number. Model is applied when both the file name is specified and **User** is specified using [Port2PdeembedCktModel Property](#).

[Learn more about S2P files.](#)

Note: This command affects ALL measurements on the channel.

VB Syntax *fixture.strPort2Pdeembed_S2PFile(port) = value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

port (**Integer**) Port number to receive circuit model.

value (**String**) Full path, file name, and extension (.s2P) of the de-embedding circuit.

Files are typically stored in "c:\users\public\network analyzer\documents

Return Type String

Default Not Applicable

Examples

```
fixture.strPort2Pdeembed_S2PFile(2) = "c:\users\public\network analyzer\documents\myFile.s2p" 'Write
```

```
value = fixture.strPort2Pdeembed_S2PFile(1) 'Read
```

C++ Syntax HRESULT get_strPort2Pdeembed_S2PFile(short port BSTR *bstrFile)

HRESULT put_strPort2Pdeembed_S2PFile(short port BSTR bstrFile)

Interface IFixturing

strPortMatch_S2PFile Property

Description Sets and returns the Port Matching 'S2P' file name for the specified port number. Model is applied when both the file name is specified and **User** is specified using [PortMatchingCktModel Property](#).

[Learn more about S2P files.](#)

Note: This command affects ALL measurements on the channel.

VB Syntax *fixture.strPortMatch_S2PFile(port) = value*

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

port (**Integer**) Port number to receive circuit model.

value (**String**) Full path, file name, and extension (.s2P) of the matching circuit.

Files are typically stored in "c:\users\public\network analyzer\documents".

Return Type String

Default Not Applicable

Examples

```
fixture.strPortMatch_S2PFile(2) = "c:\users\public\network analyzer\documents\myFile.s2p" 'Write
```

```
value = fixture.strPortMatch_S2PFile(1) 'Read
```

C++ Syntax HRESULT get_strPortMatch_S2PFile(short port BSTR *bstrFile)

HRESULT put_strPortMatch_S2PFile(short port BSTR bstrFile)

Interface IFixturing

Last Modified:

13-Nov-2013 Fixed syntax

SubPointTrigger Property

Description Enables / Disables subpoint triggering. When enabled and performing [Point Averaging](#), Each rising edge of P0 triggers a subpoint (one of N acquisitions in an N point average). Must also enable the P0 generator using [pulse.State\(n\)](#).

VB Syntax `pulse.SubPointTrigger(n) = value`

Variable [\(Type\)](#) - Description

pulse A [PulseGenerator](#) (object)

n (**Integer**) Pulse generator number. **Must be 0** as this is the generator that triggers the ADC.

value (**Boolean**) Enable or disable SubPointTrigger

- **True** - turns subpoint triggering ON.
- **False** - turns subpoint triggering OFF.

Return Type Boolean

Default False

Examples `pulse.SubPointTrigger(0) = True` **Write**

`bool = pulse.SubPointTrigger(0)` **Read**

C++ Syntax HRESULT get_SubPointTrigger (integer pulse, VARIANT_BOOL* on_off);

HRESULT put_SubPointTrigger (integer pulse, VARIANT_BOOL on_off);

Interface IPulseGenerator2

Last Modified:

20-Jul-2009 MX New topic

SupportedGenericParameters Property

Description Returns a list of supported parameters for the specified measurement class.

VB Syntax *value* = *measClassProps*.SupportedGenericParameters

Variable [\(Type\)](#) - Description

value (Variant) - Variable to store the returned array of parameters.

measClassProps A [MeasurementClassProperties](#) (Object)

Return Type Variant array

Default Not Applicable

Examples

```
'Access the MeasurementClassProperties Object

Set app = CreateObject("AgilentPNA835x.Application")

Set cap = app.Capabilities

Set measProps = cap.MeasurementClassProperties ("Swept IMD")

'Read the supported parameters

list=measProps.SupportedGenericParameters

For i = 0 To UBound(list)

    msg = msg & list(i) & vbCrLf

Next

MsgBox msg
```

C++ Syntax HRESULT get_SupportedGenericParameters(Variant *value);

Interface ICapabilities8

Last Modified:

23-May-2011 MX New topic

SupportedParameters Property

Description Returns a list of supported parameters for the specified measurement class.

VB Syntax *value* = *measClassProps*.SupportedParameters

Variable [\(Type\)](#) - Description

value (Variant) - Variable to store the returned array of parameters.

measClassProps A [MeasurementClassProperties](#) (Object)

Return Type Variant array

Default Not Applicable

Examples

```
'Access the MeasurementClassProperties Object

Set app = CreateObject("AgilentPNA835x.Application")

Set cap = app.Capabilities

Set measProps = cap.MeasurementClassProperties ("Swept IMD")

'Read the supported parameters

list=measProps.SupportedParameters

dim i

For i = 0 To UBound(list)

    msg = msg & list(i) & vbCrLf

Next

MsgBox msg
```

C++ Syntax HRESULT get_SupportedParameters(Variant *value);

Interface ICapabilities8

Last Modified:

23-May-2011 MX New topic

Sweep Delay Property

Description Specifies the time to wait just before acquisition begins for each sweep. This delay is in addition to [Dwell Time](#) and the following two External Trigger delays if enabled.

- [TriggerDelay](#) (global scope)
- [ExternalTriggerDelay](#) (channel scope)

VB Syntax *chan.SweepDelay = value*

Variable [\(Type\)](#) - Description

chan [Channel](#) (object)

value (**double**) - Sweep delay in seconds.

Return Type Double

Default 0

Examples

```
chan.SweepDelay = 3e-3 'Write
```

```
swpdelay = chan.SweepDelay 'Read
```

C++ Syntax HRESULT get_SweepDelay(double *pVal)

HRESULT put_SweepDelay(double newVal)

Interface IChannel

Last Modified:

6-Mar-2012 Fixed link

1-Mar-2010 MX New topic

SweepDwell Property

Description Sets and returns the "Dwell Before Sweep" value for an external DC Device which can be configured as either a DC Meter or a DC Source.

VB Syntax `extDC.SweepDwell = value`

Variable [\(Type\)](#) - Description

extDC An [ExternalDCDevice](#) (**object**)

value (**Double**) The dwell time (in seconds) before making a new sweep.

Return Type Double

Default 1 milliseconds

Examples `extDC.SweepDwell = 10e-3 'Write`

`dwell = extDC.SweepDwell 'Read`

C++ Syntax HRESULT get_SweepDwell (double *pValue)

HRESULT put_SweepDwell (double newVal)

Interface IExternalDCDevice

Last Modified:

15-Feb-2012 New topic

SweepEndMode Property

Description Sets and reads the event that will cause the Sweep End line to go to a low state. The line will return to a high state after the appropriate calculations are complete.

Note: This line is connected to the [HANDLER IO connector](#).

VB Syntax *object.SweepEndMode* = *value*

Variable (Type) - Description

object **(object)** - A HandlerIO or AuxIO object

value (enum as NASweepEndMode) Choose from:

0 - naSweep - the line goes low when each sweep is complete

1 - naChannelSweep - the line goes low when all the sweeps for each channel is complete.

2 - naGlobalSweep - the line goes low when all sweeps for all [triggerable](#) channels are complete.

Return Type Long Integer

Default 0 - naSweep

Examples

```
HWAuxIO.PassFailMode = naSweep 'Write
```

```
value = HWAuxIO.PassFailMode 'Read
```

C++ Syntax

```
HRESULT put_SweepEndMode ( tagNASweepEndMode Mode );
```

```
HRESULT get_SweepEndMode ( tagNASweepEndMode* Mode );
```

Interface

```
IHWAuxIO
```

```
IHWMaterialHandlerIO
```

SweepHoldOff Property

Description	Returns a boolean that represents the state of SweepHoldoff line (pin2) of the External Test Set connector.
VB Syntax	<i>value</i> = <i>ExtIO.SweepHoldOff</i>
Variable	(Type) - Description
<i>value</i>	(boolean) - Variable to store the returned data
<i>ExtIO</i>	(object) - An External IO object
Return Type	Boolean
	False - indicates the line is being held at a TTL Low
	True - indicates the line is being held at a TTL High
Default	Not Applicable
Examples	<code>value = ExtIO.SweepHoldOff</code>
C++ Syntax	HRESULT get_SweepHoldOff(VARIANT_BOOL* bValue);
Interface	IHWExternalTestSetIO

SweepOrder Property

Description Sets and returns the order number of IM products to view when [SweepType](#) = NTH is specified. This actually sets the frequency span to [DeltaF](#) * N (this command value).

VB Syntax *ims.SweepOrder = value*

Variable [\(Type\)](#) - Description

ims An [IMSpectrum](#) Object

value (Integer) Order number of IM product.

Return Type Long Integer

Default 9

Examples `ims.SweepOrder = 5 'Write`

`value = ims.SweepOrder 'Read`

C++ Syntax HRESULT get_SweepOrder(long *pVal)
HRESULT put_SweepOrder(long newVal)

Interface IIMSpectrum

Last Modified:

19-Aug-2008 MX New topic

SweepSpeedMode Property

Description	Sets and returns the sweep speed mode: FastSweep or Normal.
VB Syntax	<i>chan.SweepSpeedMode</i> = <i>value</i>
Variable	(Type) - Description
<i>chan</i>	A Channel (object)
<i>value</i>	(enum as NASweepSpeedMode) - Choose from: 0 - naSweepSpeedModeNormal - Standard PNA sweep mode 1 - naSweepSpeedModeFast - Fast sweep mode
Return Type	Enum
Default	0 - naSweepSpeedModeNormal
Examples	<pre>chan.SweepSpeedMode = naSweepSpeedModeNormal 'Write swpSpeed = chan.SweepSpeedMode 'Read</pre>
C++ Syntax	HRESULT get_SweepSpeedMode(tagNASweepSpeedModes* pVal) HRESULT put_SweepSpeedMode(tagNASweepSpeedModes newVal)
Interface	IChannel14

Last Modified:

26-Aug-2008 MX New topic

SweepGenerationMode Property

Description Sets the method used to generate a sweep: continuous ramp (analog) or discrete steps (stepped).

VB Syntax *object.SweepGenerationMode = value*

Variable [\(Type\)](#) - [Description](#)

object Channel (**object**)

or

CalSet (**object**) - Read-only property

value (enum NASweepGenerationModes) - **Choose either:**

0 - naSteppedSweep - source frequency is CONSTANT during measurement of each displayed point. More accurate than Analog. Dwell time can be set in this mode.

1 - naAnalogSweep - source frequency is continuously RAMPING during measurement of each displayed point. Faster than Stepped. Sweep time (not dwell time) can be set in this mode.

Return Type Long Integer

Default Analog

Examples

```
Chan.SweepGenerationMode = naAnalogSweep 'Write
swpgen = Chan.SweepGenerationMode 'Read
```

C++ Syntax HRESULT get_SweepGenerationMode(tagNASweepGenerationModes* pVal)
HRESULT put_SweepGenerationMode(tagNASweepGenerationModes newVal)

Interface IChannel
|CalSet3

SweepTime Property

Description Sets the Sweep time of the analyzer. If sweep time accuracy is critical, use ONLY the values that are attained using the up and down arrows next to the sweep time entry box.
[See Sweep Time.](#)

VB Syntax *object.SweepTime* = *value*

Variable (Type) - Description

object [Channel](#) (**object**)

or

[Segment](#) (**object**) first set [SweepTimeOption](#) to true.

or

[CalSet](#) (**object**) - Read-only property

value (**double**) - Sweep time in seconds. The maximum sweep time of the PNA is 86400 seconds (1 day).

To set the fastest sweep speed possible, set this value to 0.

Return Type Double

Default 0

Examples

```
chan.SweepTime = 3e-3 'Write
```

```
swptme = chan.SweepTime 'Read
```

C++ Syntax HRESULT get_SweepTime(double *pVal)
HRESULT put_SweepTime(double newVal)

Interface IChannel

CalSet3

ISegment2

SweepTimeOption Property

Description Enables the Sweep time or Dwell time to be set on individual sweep segments. This property must be set True **before** the sweep or dwell time commands are sent. Otherwise, those commands will be ignored.

VB Syntax `segs.SweepTimeOption = state`

Variable [\(Type\)](#) - Description

segs A [Segments](#) collection (**object**)

state (**boolean**)

True - Enables Sweep or Dwell time to be set independently.

False - Disables Sweep or Dwell time from being set independently.

Return Type Boolean

Default False

Examples

```
segs.SweepTimeOption = True 'Write
```

```
timeOption = SweepTimeOption 'Read
```

C++ Syntax HRESULT get_SweepTimeOption(VARIANT_BOOL *pVal)
HRESULT put_SweepTimeOption(VARIANT_BOOL newVal)

Interface ISegments3

Last modified:

9/29/06 MQQ New command

SweepType Property (IMD Opt 087)

Description Sets and returns the type of sweep for a Swept IMD measurement.

See a [list of commands](#) that are relevant for each sweep type.

VB Syntax `imd.SweepType = value`

Variable [\(Type\) - Description](#)

imd A [SweptIMD](#) Object

value (Enum as naSweepTypes) - Choose from:

- **0 - naIMDToneCWSweep** The main tone frequencies (F1 and F2) and power levels (P1 and P2) are held constant. Measurements are taken for the specified number of points.
- **1 - naIMDTonePowerSweep** The main tone frequencies are specified as either F1 and F2, or as FC and DeltaF. These frequencies are held constant while the power of each tone is varied from the Start Power to Stop Power
- **2 - naIMDToneCenterFreqSweep** Maintaining a constant tone spacing (DeltaF) and tone powers (P1 and P2), the center frequency (FC) is swept from Start to Stop, or can also be specified as Center and Span.
- **3 - naIMDDeltaFrequencySweep** The center frequency (FC) is held constant. The tone spacing is increased from Start DeltaF to Stop DeltaF.
- **4 - naIMDToneSegmentSweep** Same as FCenter sweep, except that the center frequencies for the sweep are constructed using the standard [segment sweep commands](#). (NOT valid for IMDx)
- **5 - naLOPowerSweep** All frequencies are fixed while the LO power is swept. (IMDx ONLY)

Return Type Enum

Default 2 - naIMDToneCenterFreqSweep

Examples

```
imd.SweepType = naIMDToneCWSweep 'Write
```

```
swptyp = imd.SweepType 'Read
```

C++ Syntax HRESULT get_SweepType(tagNASweepTypes* pVal)

HRESULT put_SweepType(tagNASweepTypes newVal)

Interface ISweptIMD

27-Mar-2009 Edited for IMDx

19-Aug-2008 MX New topic

SweepType Property (IMSpectrum Opt 087)

Description Sets and returns the type of sweep for an IMSpectrum measurement.

VB Syntax *ims.SweepType = value*

Variable [\(Type\)](#) - Description

ims An [IMSpectrum](#) Object

value (Enum as NAIMSSweepType) - Choose from:

- **0 - naIMSLinearSpan** When Tracking is enabled, allows tuning the Response Settings (receiver) to any values within the frequency range of the PNA. When Tracking is NOT enabled also allows setting the Stimulus (sources) to any values within the frequency range or the PNA.
- **1 - naIMSSecondOrderSpan** The receiver is tuned to view the 2nd order products (f2-f1 and f1+f2) of the main tones that are currently specified in Stimulus Settings. When Tracking is enabled, the main tones are specified in the Swept IMD channel.
- **2 - naIMSThirdOrderSpan** The receiver is tuned to view the 3rd order products (2f1 -f2 and 2f2-f1) of the main tones that are currently specified in Stimulus Settings. When Tracking is enabled, the main tones are specified in the Swept IMD channel.
- **3 - naIMSNthOrderSpan** The frequency range is set to N * DeltaF. This algorithm will NOT tune the receivers to see the EVEN order products.

Return Type Enum

Default 3 - naIMSNthOrderSpan

Examples

```
ims.SweepType = naIMSNthOrderSpan 'Write
```

```
swptyp = ims.SweepType 'Read
```

C++ Syntax HRESULT get_SweepType(tagNAIMSSweepType* pVal)
 HRESULT put_SweepType(tagNAIMSSweepType newVal)

Interface IMSpectrum

Last Modified:

19-Aug-2008 MX New topic

SweepType Property

Description Sets and returns the type of sweep. First set SweepType, then set sweep parameters such as frequency or power settings.

VB Syntax *object.SweepType = value*

Variable [\(Type\)](#) - Description

object One of the following:

- [Channel](#) (object)
- [FOMRange](#) (object) Must be an [UNCOUPLED](#) range.
- [CalSet](#) (object) - Read-only property

value (enum NASweepTypes) - Choose from:

0 - naLinearSweep

1 - naLogSweep

2 - naPowerSweep

3 - naCWTimeSweep

4 - naSegmentSweep

5 - naPhaseSweep

Note: Sweep type cannot be set to Segment sweep if there are no segments turned ON. A segment is automatically turned ON when a application is created.

Return Type Long Integer

Default naLinearSweep

Examples

```
chan.SweepType = naPowerSweep 'Write
```

```
swptyp = chan.SweepType 'Read
```

C++ Syntax HRESULT get_SweepType(tagNASweepTypes* pVal)

HRESULT put_SweepType(tagNASweepTypes newVal)

Interface IChannel

|CalSet3

IFOMRange

Last Modified:

3-Dec-2010	Added phase sweep
6-May-2010	Added sweep type note.
8-Mar-2007	Added FOMRange

SystemImpedanceZ0 Property

Description Sets and returns the impedance for the analyzer.

VB Syntax `app.SystemImpedanceZ0 = value`

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

value (double) Analyzer Impedance. Choose any number between 0 and 1000 ohms.

Return Type Double

Default 50

Examples `app.SystemImpedanceZ0 = 75 'Write`

`z0 = app.SystemImpedanceZ0 'Read`

C++ Syntax HRESULT get_SystemImpedanceZ0(double dSystemZ0)
HRESULT put_SystemImpedanceZ0(double *pdSystemZ0)

Interface IApplication

SystemName Property

Description Returns the computer name of the PNA.

VB Syntax `name = app.SystemName`

Variable [\(Type\)](#) - Description

name (String) Variable to store the returned computer name.

app An [Application](#) (**object**)

Return Type String

Default Not Applicable

Examples `name = app.SystemName`

C++ Syntax HRESULT SystemName(BSTR* computerName)

Interface IApplication

TargetValue Property

Description Sets the target value for the marker when doing Target Searches ([SearchTargetLeft](#), [SearchTarget](#), [SearchTargetRight](#)).

VB Syntax `mark.TargetValue = value`

Variable [\(Type\)](#) - Description

mark A [Marker](#) (**object**)

value (**single**) - Target value. Choose any number between: **-500** and **500**

Return Type Single

Default 0

Examples

```
mark.TargetValue = 10.5 'Write
```

```
target = mark.TargetValue 'Read
```

C++ Syntax HRESULT get_TargetValue(float *pVal)
HRESULT put_TargetValue(float newVal)

Interface IMarker

TestPortPower Property

Description Sets or returns the RF power level for the channel
or
 Sets or returns the RF power level of the segment.

VB Syntax *object*.TestPortPower(*srcPort*) = *value*

Variable [\(Type\)](#) - Description

object A [Channel](#) (**object**) - to set coupled power, use chan.[CouplePorts](#). If CouplePorts = False, then each port power can be set independently. Otherwise, chanTestPortPower (1) = value sets power level at both ports.

or

A [CalSet](#) (**object**)

or

A [Segment](#) (**object**)

srcPort (**long integer**) - Source Port number.

Note: If the port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-x model, then you must use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#).

value (**double**) - RF Power in dBm.

Note: The range of settable power values depends on the PNA model and if source attenuators are installed. To determine the range of values, use [cap.MaximumSourceALCPower](#) and [cap.MinimumSourceALCPower](#)

Actual achievable leveled power depends on frequency.

Return Type Double

Default 0

Examples

```
chan.TestPortPower(1) = 5 'sets the port 1 RF power level for the channel object -Write
```

```
powerlev = Chan.TestPortPower(1) 'Read
```

C++ Syntax HRESULT get_TestPortPower(long port, double *pVal)
 HRESULT put_TestPortPower(long port, double newVal)

Interface IChannel
ICalSet3
ISegment

Last Modified:

24-Apr-2008 Added note for string names
30-Apr-2007 Edited for src strings

TestSetType Property

Description	Returns the Test Set Type (model) that was used for the Cal Set.
VB Syntax	<i>TSType</i> = <i>calset</i> . TestSetType
Variable	(Type) - Description
<i>TSType</i>	(String) Variable to store the returned test set model.
<i>calset</i>	A Cal Set object.
Return Type	String
Default	Depends on the test set.
Example	<code>TSType = calset.TestSetType</code>
C++ Syntax	HRESULT get_OutputPorts(BSTR *mapping);
Interface	ICalset5

Last modified:

9/18/06 MQ Added for multiport

Text Property

Description Specifies an equation or expression to be used on the measurement.

VB Syntax *eq.Text* = *eqText*

Variable [\(Type\)](#) - Description

eq [Equation](#) (**object**)

eqText (**String**) - Any valid equation or expression.

Return Type String

Default Not Applicable

Examples `eq.Text = "foo=S11/S21"`

`equation = eq.Text 'Read`

C++ Syntax HRESULT get_Text(BSTR *equation)
HRESULT put_Text(BSTR equation)

Interface IEquation

ThruCalMethod Property - Superseded

Description	This command is replaced by PathThruMethod Property . Sets and returns the method for performing the thru portion of the calibration.
VB Syntax	<i>obj</i> . ThruCalMethod = <i>value</i>
Variable	(Type) - Description
<i>obj</i>	SMCType (object) or VMCType (object)
<i>value</i>	(String) Specifies the Thru method. Case insensitive - include spaces. Choose from: "Default" "Flush Thru" or "FLUSH" "Unknown Thru" or "UNKN" "Adapter Removal" or "ADAP"
Return Type	String
Default	Default
Examples	<pre>SMC.ThruCalMethod = "UNKN"</pre>
C++ Syntax	<pre>HRESULT put_ThruCalMethod(enum NAThruCalMethod thruMethod); HRESULT get_ThruCalMethod(enum NAThruCalMethod *thruMethod);</pre>
Interface	SMCType VMCType

ThruCalMethod Property **Superseded**

Description This command is replaced by PathThruMethod Property .

Sets and returns the method for performing the Cal Method and the THRU portion of the calibration.

VB Syntax *guidedCal* .**ThruCalMethod** = *value*

Variable (Type) - Description

guidedCal GuidedCalibration (object)

value (Enum as NATHruCalMethod) Choose from:

0 - naDefaultCalMethod - allow the PNA to choose the best possible method (from the following) depending on whether the device or ECal module is insertable or non-insertable and given the model number of the PNA. (default selection if omitted.)

1 - naAdapterRemoval - Perform Adapter removal calibration.

2 - naFlushThru - Perform Flush Thru calibration.

3 - naDefinedThru - Perform Defined Thru calibration. If performing an ECal, this is the Thru standard in the ECal module.

4 - naUnknownThru - Perform Unknown Thru calibration.

5 - naSOLT - Perform SOLT calibration

6 - naTRL - Perform TRL calibration

7 - naQSOLT - Perform QSOLT calibration.

Learn more about Cal Methods

Return Type Enum

Default 0 - naDefaultCalMethod

Examples `guided.ThruCalMethod = naDefinedThru`

C++ Syntax HRESULT get_ThruCalMethod(enum NATHruCalMethod *thruMethod);

HRESULT put_ThruCalMethod(enum NATHruCalMethod

thruMethod);

Interface IGuidedCalibration

Last Modified:

30-Apr-2007 MX Superseded

ThruPortList Property

Description	<p>Note: Available only on PNA releases 5.0 and greater.</p> <p>Note: Do NOT send this command to rely on SmartCal to determine the most accurate Thru port pairs for the cal. You can send the query form of this command to learn the port pairs determined by SmartCal.</p> <p>Sets and returns the Thru connection port pairs for the calibration. Send the query form of this command to learn the Thru pairs determined by SmartCal.</p> <p>See Thru Pairs Sequence to learn how to send this and other Thru commands.</p> <p>Learn more about Thru method and port pairings.</p> <p>See an example of a 4-port guided calibration using COM.</p>
VB Syntax	<code>guidedCal.ThruPortList = t1a, t1b, t2a, t2b, t3a, t3b</code>
Variable	(Type) - Description
<code>guidedCal</code>	GuidedCalibration (object)
<code>t1a, t1b...</code>	(Variant) Port numbers in pairs - a one-dimensional array of Long integers.
	t1a, t1b (Thru1 - port A and port B)
	t2a, t2b (Thru2 - port A and port B)
	t3a, t3b (Thru3 - port A and port B)
Return Type	Variant - a one-dimensional array of Long integers.
Default	The most accurate port pairs for the cal.
Example	<pre>thruList = Array(1,2,1,3,1,4) guided.ThruPortList = thruList</pre> <p>'Sets the following three thru connections for a 4-port calibration: Thru 1 - ports 1 and 2 Thru 2 - ports 1 and 3 Thru 3 - ports 1 and 4</p>
C++ Syntax	<pre>HRESULT get_ThruPortList(VARIANT* portList); HRESULT put_ThruPortList(VARIANT portList);</pre>

Interface IGuidedCalibration

TimeOut Property

Description Sets and returns the Time out value for communication with the external device. An error is returned if communication with the device is not successful within this period of time.

VB Syntax *extDevices*.**TimeOut** = *value*

Variable [\(Type\)](#) - [Description](#)

extDevices An [ExternalDevice](#) (**object**)

value (Double) Time out in milliseconds.

Return Type Double

Default 20000 milliseconds (20 seconds)

Examples `extDevices.TimeOut = 1000 Write`

`value = extDevices.TimeOut 'Read`

C++ Syntax HRESULT get_TimeOut(Double* value);

HRESULT put_TimeOut(Double newVal);

Interface IExternalDevices

Last Modified:

31-Jul-2009 MX New topic

Title Property

Description Writes or reads a custom title for the window. Newer entries replace (not append) older entries. Turn the title ON and OFF with [TitleState](#)

VB Syntax `win.Title = string`

Variable [\(Type\)](#) - Description

win A NaWindow (**object**)

string (**long**) - Title limited to 50 characters.

**Return
Type** String

Default Null

Examples `win.Title = "Hello World" 'Write`

`titl = win.Title 'Read`

C++ Syntax HRESULT get_Title(BSTR *title)
HRESULT put_Title(BSTR title)

Interface INAWindow

TitleState Property

Description Turns ON and OFF the window title. Write a window title with [Title](#)

VB Syntax `win.TitleState = state`

Variable [\(Type\)](#) - Description

win A NaWindow (**object**)

state (**boolean**)

True - Title ON

False - Title OFF

Return Type Boolean

Default False

Examples `win.TitleState = True 'Write`

`titlestate = win.TitleState 'Read`

C++ Syntax HRESULT get_TitleState(VARIANT_BOOL* bState)
HRESULT put_TitleState(VARIANT_BOOL bState)

Interface INAWindow

Tolerance Property

Description Sets and returns the tolerance value for leveling sweeps.

VB Syntax *RxLevel.Tolerance(srcPort) = value*

Variable [\(Type\)](#) - Description

RxLevel A [ReceiverLeveling](#) Object

srcPort (Long Integer) Source port for which to set the tolerance value for Receiver Leveling.

Note: If the source port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-X model, then use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#)

value (Double) Tolerance level in dB.

Return Type (Double)

Default .1 dB

Examples

```
rxLevel.Tolerance (1) = .5 ' Write  
value = rxLevel.Tolerance 2 ' Read
```

C++ Syntax

```
HRESULT get_Tolerance(long port, double* pVal);
```

```
HRESULT put_Tolerance(long port, double newVal);
```

Interface IReceiverLevelingConfiguration

Last Modified:

13-Feb-2009 MX New topic

TonePower Property

Description Sets and returns the power level of the Main Tones. Use with IMD sweep types:

- **naIMDToneCWSweep**,
- **naIMDToneCenterFreqSweep**
- **naIMDDeltaFrequencySweep**.

When tone power is coupled, setting either F1 or F2 power sets both.

VB Syntax *object*.**TonePower** (*tone*) = *value*

Variable [\(Type\)](#) - Description

object A [SweptIMD](#) or [IMSpectrum](#) Object

tone (Enum as NAIMDTonePowerID) Choose from:

0 - naIMDF1Power - F1 tone

1 - naIMDF2Power - F2 tone

value (Double) Tone power level in dBm. Choose a value between +30 dBm and -30 dBm.

Return Type Double

Default -20 dBm

Examples

```
imd.TonePower(naIMDF1Power) = 0 'Write
```

```
value = imd.TonePower(naIMDF2Power) 'Read
```

C++ Syntax HRESULT get_TonePower(tagNAIMDTonePowerID naIMDF1, double * pVal);
 HRESULT put_TonePower(tagNAIMDTonePowerID naIMDF1, double newVal);

Interface ISweptIMD

IIMSpectrum

Last Modified:

19-Aug-2008 MX New topic

TonePowerSetAt Property - Superseded

Description **Note:** This command is replaced with [LevelingMethod Property](#)

Sets and returns whether tone power is specified at the DUT input or output.

VB Syntax *object.TonePowerSetAt = value***Variable** [\(Type\)](#) - Description*object* A [SweptIMD](#) Object or [IMSpectrum](#) Object*value* (Enum as NAPortMode)**0 - naInput** - Specified power level is set at the DUT input.**1 - naOutput** - Specified power level is set at the DUT output.**Return Type** Enum**Default** **0 - naInput****Examples** `imd.TonePowerSetAt = naOutput 'Write``value = ims.TonePowerSetAt 'Read`**C++ Syntax** HRESULT get_TonePowerSetAt(tagNAPortMode *pVal)

HRESULT put_TonePowerSetAt(tagNAPortMode pVal)

Interface ISweptIMD2IImSpectrum2

Last modified:

14-Nov-2012 Superseded

5-May-2011 New topic

TonePowerStart Property

Description Sets and returns the start power level of the Main tones. Use with IMD sweep type=
naIMDTonePowerSweep.

When tone power is coupled, setting either F1 or F2 power sets both.

VB Syntax *object*.**TonePowerStart** (*tone*) = *value*

Variable [\(Type\)](#) - Description

object A [SweptIMD](#) or [IIMSpectrum](#) Object

tone (Enum as NAIMDTonePowerID) Choose from:

0 - naIMDF1Power - F1 tone

1 - naIMDF2Power - F2 tone

value (Double) Start power in dBm. Choose a value between +30 dBm and -30 dBm.

Return Type Double

Default -20 dBm

Examples `imd.TonePowerStart(naIMDF1Power) = 0 'Write`

`value = imd.TonePowerStart(naIMDF2Power) 'Read`

C++ Syntax HRESULT get_TonePowerStart(tagNAIMDTonePowerID naIMDF1, double * pVal);
HRESULT put_TonePowerStart(tagNAIMDTonePowerID naIMDF1, double newVal);

Interface ISweptIMD
IIMSpectrum

Last Modified:

19-Aug-2008 MX New topic

TonePowerStop Property

Description Sets and returns the stop power level of the Main tones. Use with IMD sweep type=**naIMDTonePowerSweep**.

When tone power is coupled, setting either F1 or F2 power sets both.

VB Syntax *object*.**TonePowerStop** (*tone*) = *value*

Variable [\(Type\)](#) - Description

object A [SweptIMD](#) or [IMSpectrum](#) Object

tone (Enum as NAIMDTonePowerID) Choose from:

0 - naIMDF1Power - F1 tone

1 - naIMDF2Power - F2 tone

value (Double) Stop power in dBm. Choose a value between +30 dBm and -30 dBm.

Return Type Double

Default -20 dBm

Examples `imd.TonePowerStop (naIMDF1Power) = 0 'Write`

`value = imd.TonePowerStop (naIMDF2Power) 'Read`

C++ Syntax HRESULT get_TonePowerStop (tagNAIMDTonePowerID naIMDF1, double * pVal);
 HRESULT put_TonePowerStop (tagNAIMDTonePowerID naIMDF1, double newVal);

Interface ISweptIMD
 IIMSpectrum

Last Modified:

19-Aug-2008 MX New topic

TotallIterations Property

Description	Returns the total number of iterations required by the last SMART sweep. Returns number of power points for a 2D sweep.
VB Syntax	<i>iter</i> = <i>gca</i> . TotallIterations
Variable	(Type) - Description
<i>iter</i>	(Integer) Variable to store the returned number of iterations.
<i>gca</i>	A GainCompression (object)
Return Type	Integer
Default	Not Applicable
Examples	<pre>data = gca.TotalIterations</pre>
C++ Syntax	HRESULT get_TotalIterations();
Interface	IGainCompression2

Last Modified:

9-May-2008 MX New topic

TotalNumberOfPoints Property

Description Read the total number of points a complete GCA measurement will generate.

- For 2D modes, this is Frequency * Power points
- For SMART Sweep, this is Frequency points.

The total can NOT exceed the [PNA maximum](#).

See [Frequency](#) and [Power](#) points.

VB Syntax *value* = *gca*.**TotalNumberOfPoints**

Variable [\(Type\)](#) - Description

value (**integer**) Variable to store the returned total number of points

gca A [GainCompression](#) (**object**)

Return Type Integer

Default 5226 (201 * 26)

Example `totPoints = gca.TotalNumberOfPoints 'Read`

C++ Syntax HRESULT get_TotalNumberOfPoints(int* pVal)

Interface IGainCompression

Last Modified:

11-Sep-2007 MX New topic

Touchscreen Property

Description Sets and reads the state of the PNA-X Touchscreen (ON and OFF).
This setting remains until changed again from the front-panel or remote command.

VB Syntax *app*.**Touchscreen** = *state*

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

state (boolean)

False (0) - Disables use of Touchscreen

True (1) - Enables use of Touchscreen

Return Type Boolean

False - OFF

True - ON

Default **TRUE** when shipped from factory.

Examples

```
app.Touchscreen = True 'Write  
coupl = app.Touchscreen 'Read
```

C++ Syntax HRESULT put_Touchscreen(VARIANT_BOOL bState)

HRESULT get_Touchscreen(VARIANT_BOOL *bState)

Interface IApplication12

Last Modified:

23-Feb-2007 MX New topic

TraceMath Property

Description Performs math operations on the measurement object and the trace stored in memory. (There MUST be a trace stored in Memory to perform math. See [Meas.DataToMemory](#) method.)

VB Syntax *meas.TraceMath = value*

Variable [\(Type\)](#) - Description

meas A [Measurement](#) (object)

value (enum **NAMathOperation**) - Choose from:

0 - naDataNormal

1 - naDataMinusMemory

2 - naDataPlusMemory

3 - naDataDivMemory

4 - naDataTimesMemory

Return Type NAMathOperation

Default Normal (0)

Examples

```
meas.TraceMath = naDataMinusMemory 'Write
```

```
mathOperation = meas.TraceMath 'Read
```

C++ Syntax HRESULT get_TraceMath(tagNAMathOperation* pMathOp)
HRESULT put_TraceMath(tagNAMathOperation mathOp)

Interface IMeasurement

TraceMax Property

Description Maximizes (isolates) or restores the active trace in the active window. When TraceMax is ON, the active trace is the **ONLY** trace on the display. All other traces are hidden.

VB Syntax `meas.TraceMax = state`

Variable [\(Type\)](#) - Description

meas A [Measurement](#) (object)

state (boolean) - Choose from:

True - Maximizes / isolates the active trace in the window.

False - Restores other traces to be viewed in the window.

Return Type Boolean

Default False

Examples `meas.TraceMax = True`

`state = meas.TraceMax`

C++ Syntax HRESULT get_TraceMax(VARIANT_BOOL bState)
HRESULT put_TraceMax(VARIANT_BOOL* bState)

Interface IMeasurement10

Last Modified:

26-Aug-2008 MX New topic

TraceTitle Property

Description	Writes and reads data for the trace title area. The trace title is embedded in the trace status field. Learn more about Trace Titles The title is turned ON and OFF using TraceTitleState .
VB Syntax	<i>meas</i> . TraceTitle = <i>value</i>
Variable	(Type) - Description
<i>meas</i>	A Measurement (object)
<i>value</i>	(string) - Title to be displayed. Any characters (no spaces), enclosed with quotes.
Return Type	String
Default	Not Applicable
Examples	<pre>meas.TraceTitle = "My new s11 measurement"</pre> <pre>title = TraceTitle 'Read'</pre>
C++ Syntax	HRESULT get_TraceTitle(BSTR *title); HRESULT put_TraceTitle(BSTR title);
Interface	IMeasurement8

Last Modified:

16-Jan-2007 MX New topic

TraceTitleState Property

Description Turns display of the Trace Title ON or OFF. When turned OFF, the previous trace title returns. Create a trace title using [TraceTitle Property](#)

VB Syntax *meas.TraceTitleState = value*

Variable [\(Type\)](#) - Description

meas A [Measurement \(object\)](#)

value (**boolean**) - Choose from:

True - Turns the trace title ON

False - Turns the trace title OFF

Return Type Boolean

Default False

Examples

```
meas.TraceTitleState = False
```

```
title = TraceTitleState 'Read
```

C++ Syntax HRESULT get_TraceTitleState(VARIANT_BOOL *isTitleON);
HRESULT put_TraceTitleState(VARIANT_BOOL isTitleON);

Interface IMeasurement8

Last Modified:

18-Jun-2007 MX New topic

Tracking Property

Description This property, when on, executes the search function ([marker.SearchFunction](#)) every sweep.

In effect, turning Tracking ON is the same as executing one of the immediate, one-time, "Search..." methods (such as [SearchMin](#), [SearchMax](#)) for every sweep.

VB Syntax `mark.Tracking = state`

Variable [\(Type\) - Description](#)

mark A [Marker](#) (object)

state (**boolean**) - Tracking state. Choose from:

False - Tracking OFF

True - Tracking ON

Return Type Boolean

Default False

Examples

```
mark.Tracking = False 'Write
```

```
markTracking = mark.Type 'Read
```

C++ Syntax HRESULT put_Tracking(VARIANT_BOOL bOn)
HRESULT get_Tracking(VARIANT_BOOL * pbOn)

Interface IMarker

TrackingChannel Property

Description Sets and returns the IMD channel number to which the IM Spectrum channel is coupled. Use [TrackingEnable](#) to enable tracking.

VB Syntax *ims.TrackingChannel = value*

Variable [\(Type\)](#) - Description

ims An [IMSpectrum](#) Object

value (Integer) Existing IMD channel number to which frequency and power settings are coupled.

Return Type Long Integer

Default First existing IMD channel

Examples

```
ims.TrackingChannel = 1 'Write
```

```
value = ims.TrackingChannel 'Read
```

C++ Syntax HRESULT get_TrackingChannel(long *pVal)
HRESULT put_TrackingChannel(long newVal)

Interface IIMSpectrum

Last Modified:

19-Aug-2008 MX New topic

TrackingEnable Property

Description When an IMD channel exists, allows the IM Spectrum frequency and power setting to track (couple with) the IMD channel settings. Use [TrackingChannel](#) to set the channel number to track.

VB Syntax *ims*.**TrackingEnable** = *value*

Variable [\(Type\)](#) - Description

ims An [IMSpectrum](#) Object

value (Boolean) Tracking state. Choose from:

True - IM Spectrum frequency and power settings track the IMD channel settings.

False - IM Spectrum frequency and power settings are specified in the IMS channel.

Return Type Boolean

Default False

Examples

```
ims.TrackingEnable = True 'Write
```

```
value = ims.TrackingEnable 'Read
```

C++ Syntax HRESULT get_TrackingEnable(VARIANT_BOOL* bValue)
HRESULT put_TrackingEnable(VARIANT_BOOL newVal)

Interface IIMSpectrum

Last Modified:

19-Aug-2008 MX New topic

TrackingManualStepEnable Property

Description Sets and returns the step sweep mode for the IM Spectrum channel.

VB Syntax *ims*.TrackingManualStepEnable = *value*

Variable [\(Type\)](#) - Description

ims An [IMSpectrum](#) Object

value (Boolean) Choose from:

False - Automatic Step

True - Manual Step

Return Type Boolean

Default **False** - Automatic Step

Examples `ims.TrackingManualStepEnable = True 'Write`

`value = ims.TrackingManualStepEnable 'Read`

C++ Syntax HRESULT get_TrackingManualStepEnable(VARIANT_BOOL *bVal)
HRESULT put_TrackingManualStepEnable(VARIANT_BOOL newVal)

Interface IIMSpectrum

Last Modified:

19-Aug-2008 MX New topic

TrackingStepIndex Property

Description When [TrackingManualStepEnable](#) = True (Manual step), sets and returns the data point number at which the IM spectrum measurement occurs.

VB Syntax *ims.TrackingStepIndex* = *value*

Variable [\(Type\)](#) - Description

ims An [IMSpectrum](#) Object

value (Integer) Existing IMD channel number to which frequency and power settings are coupled.

Return Type Long Integer

Default 1

Examples `ims.TrackingStepIndex = 50 'Write`

`value = ims.TrackingStepIndex 'Read`

C++ Syntax HRESULT get_TrackingStepIndex(long *pVal)
HRESULT put_TrackingStepIndex(long newVal)

Interface IIMSpectrum

Last Modified:

19-Aug-2008 MX New topic

TriggerDelay Property

Description Sets and reads the trigger delay for all measurements (GLOBAL). This delay is only applied while in [app.Source = naTriggerSourceExternal](#) and [trigsetup.Scope = naGlobalTrigger](#) . After an external trigger is applied, the start of the sweep is delayed for the specified delay value plus any inherent latency.

To apply a trigger delay for a channel only, use [ExternalTriggerDelay Property](#).

VB Syntax `app.TriggerDelay = value`

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

value **Double**- Trigger delay value in seconds. Range is from 0 to 107

Return Type Double

Default 0

Examples

```
app.TriggerDelay = .003 'Write
```

```
delay = app.TriggerDelay 'Read
```

C++ Syntax HRESULT get_TriggerDelay(double *delay);

HRESULT put_TriggerDelay(double delay)

Interface IApplication

TriggerInPolarity Property

Description Specifies the polarity of the trigger IN signal.

- **AuxTrigger Object** - Sets the polarity to which the rear-panel AuxTrig IN responds.
- **PulseGenerator Object** - Sets the polarity of trigger to which the internal pulse generators will respond when being externally triggered. **Note:** This feature requires DSP version: **4.0 FPGA:34** or higher. [Learn more](#).

Note: Used on PNA-X ONLY.

VB Syntax *object.TriggerInPolarity = value*

Variable [\(Type\)](#) - Description

object An [AuxTrigger](#) (object) or

A [PulseGenerator](#) (object)

value (**enum NATriggerPolarity**) - Choose from:

naTriggerPositive PNA responds to rising edge or HIGH level

naTriggerNegative PNA responds to falling edge or LOW level.

Set Edge or Level triggering using [TriggerInType Property](#)

Return Type Enum

Default AuxTriggerIn Object - naTriggerNegative

PulseGenerator Object - naTriggerPositive. Also the polarity used when the PNA-X does not have the required DSP hardware

Examples `obj.TriggerInPolarity = naTriggerPositive 'Write`

`value = obj.TriggerInPolarity 'Read the value`

C++ Syntax HRESULT get_TriggerInPolarity(enum NATriggerPolarity *val);
HRESULT put_TriggerInPolarity(enum NATriggerPolarity val);

Interface IAuxTrigger

IPulseGenerator3

9-Dec-2009 Added PulseGenerator Object
5-Sep-2008 Added Note
14-Dec-2006 MX New topic

TriggerInType Property

Description Specifies the type of trigger input being supplied to the PNA.

- **AuxTrigger Object** - Sets the type to which the rear-panel AuxTrig IN responds.
- **PulseGenerator Object** - Sets the type of trigger to which the internal pulse generators will respond when being externally triggered. **Note:** This feature requires DSP version: **4.0 FPGA:34** or higher. [Learn more](#).

Note: Use on PNA-X ONLY.

VB Syntax *obj.TriggerInType = value*

Variable [\(Type\)](#) - Description

obj An [AuxTrigger](#) (object) or

A [PulseGenerator](#) (object)

value (enum NATriggerSignalType) Choose from:

naTriggerEdge PNA responds to the edge (rising or falling) of a signal

naTriggerLevel PNA responds to the level (HIGH or LOW) of a signal

Use [TriggerInPolarity](#) to set Positive or Negative polarity.

Return Type Enum

Default **naTriggerLevel** - Also the type used for the PulseGenerator Object when the PNA-X does not have the required DSP hardware

Examples

```
obj.TriggerInType = naTriggerEdge 'Write
```

```
value = obj.TriggerInType 'Read the value
```

C++ Syntax HRESULT get_TriggerInType(enum NATriggerSignalType *val);
HRESULT put_TriggerInType(enum NATriggerSignalType val);

Interface IAuxTrigger

IPulseGenerator3

9-Dec-2009 Added PulseGenerator obj.
5-Sep-2008 Added Note
14-Dec-2006 MX New topic

TriggerMode (ExtendedProperties) Property

Description Sets and returns the trigger mode for an external source.

VB Syntax `extSource.TriggerMode = value`

Variable [\(Type\)](#) - Description

extSource An [ExternalSource Object](#) (object)

value (enum NAExtDevTriggerMode) - Choose from:

0 - naExtDevTriggerModeCW

1 - naExtDevTriggerModeHW

Return Type Enum

Default **0 - naExtDevTriggerModeCW**

Examples `extSource.TriggerMode = naExtDevTriggerModeCW 'Write`

`tm = extSource.TriggerMode 'Read`

C++ Syntax HRESULT get_TriggerMode (tagNAExtDevTriggerMode *pMode)
HRESULT put_TriggerMode (tagNAExtDevTriggerMode newMode)

Interface IExternalSource

Last Modified:

8-Sep-2009 MX New topic

TriggerOutDuration Property

Description Specifies the width of the output pulse, which is the time that the Aux trigger output will be asserted.

VB Syntax *auxTrig.TriggerOutDuration = value*

Variable [\(Type\)](#) - Description

auxTrig An [AuxTrigger](#) (object)

value (**single**) - Duration value in seconds. Choose a value between 1E-6 and 1.

Return Type Double

Default 1E-6 sec

Examples `auxTrig.TriggerOutDuration = 1e-3 'Write`

`value = auxTrig.TriggerOutDuration 'Read the value`

C++ Syntax HRESULT get_TriggerOutDuration(double *val);
HRESULT put_TriggerOutDuration(double val);

Interface IAuxTrigger

Last Modified:

14-Dec-2006 MX New topic

TriggerOutInterval Property

Description Specifies how often a trigger output signal is sent.

VB Syntax `auxTrig.TriggerOutInterval = value`

Variable [\(Type\)](#) - Description

auxTrig An [AuxTrigger](#) (object)

value (single) - Choose from:

0 - naTriggerModePoint - a single data point is measured with each trigger signal the channel receives. Subsequent trigger signals continue to go to the channel in Point mode until the channel measurements are complete. This is effectively the same as [trigger point mode](#).

1 - naTriggerModeMeasurement - entire traces are swept with a trigger signal. which and how many traces depends on the Scope setting.

Return Type Enum

Default 1 - naTriggerModeMeasurement

Exaamples

```
auxTrig.TriggerOutInterval = naTriggerModeMeasurement 'Write  
value = auxTrig.TriggerOutInterval 'Read the value
```

C++ Syntax HRESULT get_TriggerOutInterval(enum NATriggerMode *val);
HRESULT put_TriggerOutInterval(enum NATriggerMode val);

Interface IAuxTrigger

Last Modified:

14-Dec-2006 MX New topic

TriggerOutPolarity Property

Description Specifies the polarity of the trigger output signal being supplied by the PNA.

VB Syntax `auxTrig.TriggerOutPolarity = value`

Variable [\(Type\)](#) - Description

auxTrig An [AuxTrigger](#) (object)

value (enum **NATriggerPolarity**) - Choose from:

naTriggerPositive PNA sends positive going (active HIGH) pulse.

naTriggerNegative PNA sends negative going (active LOW) pulse.

Return Type Enum

Default

Exaamples `auxTrig.TriggerOutPolarity = naTriggerPositive 'Write`

`value = auxTrig.TriggerOutPolarity 'Read the value`

C++ Syntax `HRESULT get_TriggerOutPolarity(enum NATriggerPolarity *val);`
`HRESULT put_TriggerOutPolarity(enum NATriggerPolarity val);`

Interface IAuxTrigger

Last Modified:

14-Dec-2006 MX New topic

TriggerOutPosition Property

Description Specifies whether the Aux trigger out signal is sent Before or After the acquisition.

VB Syntax `auxTrig.TriggerOutPosition = value`

Variable [\(Type\)](#) - Description

auxTrig An [AuxTrigger](#) (object)

value (enum NATriggerPosition) Choose from:

naTriggerOutBeforeAcquire Use if the external device needs to be triggered before the data is acquired, such as a power meter.

naTriggerOutAfterAcquire Use if the external device needs to be triggered just after data has been acquired, such as an external source. This could be more efficient since it allows the external device to get ready for the next acquisition at the same time as the PNA.

Return Type Enum

Default **naTriggerOutAfterAcquire**

Examples `auxTrig.TriggerOutPosition = naTriggerOutAfterAcquire 'Write`

`value = auxTrig.TriggerOutPosition 'Read the value`

C++ Syntax HRESULT get_TriggerOutPosition(enum NATriggerPosition *val);
HRESULT put_TriggerOutPosition(NATriggerPosition val);

Interface IAuxTrigger

Last Modified:

14-Dec-2006 MX New topic

TriggerOutputEnabled Property - **Superseded**

Description	Use AUX.Enable to enable AUXI/O triggering. Use Trigger Source= External to enable Meas Trig Ready output. Enables the PNA to send trigger signals out the rear-panel TRIGGER OUT connector.
VB Syntax	<i>trigsetup</i> . TriggerOutputEnabled = <i>boolean</i>
Variable	(Type) - Description
<i>trigsetup</i>	A TriggerSetup2 (object)
<i>boolean</i>	Choose from: False - PNA does NOT send output trigger signals. True - PNA sends output trigger signals.
Return Type	Boolean
Default	False
Examples	<pre>trigsetup.TriggerOutputEnabled = True 'Write</pre> <pre>atba = trigsetup.TriggerOutputEnabled 'Read</pre>
C++ Syntax	HRESULT get_TriggerOutputEnabled(BOOL *pVal); HRESULT put_TriggerOutputEnabled(BOOL newVal);
Interface	ITriggerSetup2

Last modified:

13-Apr-2012 Superseded by new commands (9.50)

TriggerPort Property

Description Sets and returns the PNA port through which an external source is to be triggered.

VB Syntax *extSource*.**TriggerPort** = *value*

Variable [\(Type\)](#) - Description

extSource An [ExternalSource Object](#) (object)

value (enum **NAExtDevTriggerPort**) - Choose from:

0 - naExtDevTriggerPortBNC1 (PNA 'C' models)

1 - naExtDevTriggerPortAux1 (PNA-X models)

2 - naExtDevTriggerPortAux2 (PNA-X models)

Return Type Enum

Default For PNA 'C' models - **BNC1**

For PNA-X models - **Aux1**

Examples

```
extSource.TriggerPort = 1 'Write
```

```
trigpt = extSource.TriggerPort 'Read
```

C++ Syntax HRESULT get_TriggerPort (tagNAExtDevTriggerPort *pValue)
HRESULT put_TriggerPort (tagNAExtDevTriggerPort newVal)

Interface IExternalSource

Last Modified:

31-Jul-2009 MX New topic

TuningIFBW Property

Description Set the IF Bandwidth for Broadband and Precise tuning sweeps.

VB Syntax *obj*.**TuningIFBW** = *value*

Variable [\(Type\)](#) - Description

obj An [EmbeddedLO](#) (**object**) or

A [ConverterEmbeddedLO](#) (**object**)

value (**Double**) IF Bandwidth

Return Type (**Double**)

Default 30 kHz

Examples `embedLO.TuningIFBW = 10e3 'write`

`value = embedLO.TuningIFBW 'read`

C++ Syntax HRESULT get_TuningIFBW(double* ifbw);

HRESULT put_TuningIFBW(double ifbw);

Interface IEmbeddedLO

Last Modified:

12-Aug-2009 Added ConvEmbedLO object

18-Apr-2007 MX New topic

TuningMode Property

Description Sets and returns the method used to determine the embedded LO Frequency.

VB Syntax *obj*.**TuningMode** = *value*

Variable [\(Type\)](#) - Description

obj An [EmbeddedLO](#) (**object**) or

A [ConverterEmbeddedLO](#) (**object**)

value (**Enum as NAEmbeddedLOTuningMode**)

Tuning mode. Choose from:

0 - naEmbeddedLOTuningMode_Broadband_And_Precise

1 - naEmbeddedLOTuningMode_Precise_Only

2 - naEmbeddedLOTuningMode_None

Return Type (**Enum**)

Default 0 - naEmbeddedLOTuningMode_Broadband_And_Precise

Examples `embedLO.TuningMode = naEmbeddedLOTuningMode_None 'write`

`value = embedLO.TuningMode 'read`

C++ Syntax HRESULT get_TuningMode(enum NAEmbeddedLOTuningMode* mode);

HRESULT put_TuningMode(enum NAEmbeddedLOTuningMode mode);

Interface IEmbeddedLO

Last Modified:

12-Aug-2009 Added ConvEmbedLO object

18-Apr-2007 MX New topic

TuningSweepInterval Property

Description Set how often a tuning sweep is performed.

VB Syntax *obj*.**TuningSweepInterval** = *value*

Variable [\(Type\)](#) - Description

obj An [EmbeddedLO](#) (**object**) or

A [ConverterEmbeddedLO](#) (**object**)

value (**Long**) Tuning sweep interval.

Return Type (**Long**)

Default 1

Examples

```
embedLO.TuningSweepInterval = 3 'write .. tuning is performed  
every third measurement sweep
```

```
value = embedLO.TuningSweepInterval 'read
```

C++ Syntax HRESULT get_TuningSweepInterval(long* interval);

HRESULT put_TuningSweepInterval(long interval);

Interface IEmbeddedLO

Last Modified:

12-Aug-2009 Added ConvEmbedLO object

18-Apr-2007 MX New topic

TwoPointGroupDelayAperture Property

Description Sets the default group delay aperture setting.

VB Syntax *pref*.TwoPointGroupDelayAperture = *value*

Variable [\(Type\)](#) - Description

pref A [Preferences](#) (object)

value (**Boolean**) - Choose from:

True - Set the default group delay aperture setting to two points.

False - Set the default group delay aperture setting to 11 points.

Return Type Boolean

Default False

Examples `pref.TwoPointGroupDelayAperture = True 'Write`

`gda = pref.TwoPointGroupDelayAperture 'Read`

C++ Syntax HRESULT get_TwoPointGroupDelayAperture(VARIANT_BOOL* pVal);
HRESULT put_TwoPointGroupDelayAperture(VARIANT_BOOL pVal);

Interface IPreferences11

Last Modified:

1-Mar-2010 MX New topic

TriggerMode Property

Description These settings determine what EACH signal will trigger.

Note: Setting Point and EverySweep mode forces [Trigger.Scope](#) = `naChannelTrigger`.

VB Syntax `chan.TriggerMode = value`

Variable [\(Type\)](#) - Description

chan A [Channel](#) (object)

value (enum `NATriggerMode`) - Choose from:

0 - `naTriggerModePoint` - Each Manual or External trigger signal causes one data point to be measured.

1 - `naTriggerModeMeasurement` (**superseded** - still works but replaced with a more descriptive enum)

1 - `naTriggerModeChannel` - Each trigger signal causes **ALL traces** in that channel to be swept.

2 - `naTriggerModeEverySweep` - Each Manual or External trigger signal causes **ALL traces that share a source port** to be swept.

3 - `naTriggerModeTrace` - Allowed ONLY when [PointSweepState](#) is enabled. Each trigger signal causes two identical measurements to be triggered separately - one trigger signal is required for each measurement. Other trigger mode settings cause two identical parameters to be measured simultaneously.

Return Type Long Integer

Default **1 - `naTriggerModeChannel`**

Examples `chan.TriggerMode = naTriggerModePoint 'Write`

`trigtyp = chan.TriggerMode 'Read`

C++ Syntax `HRESULT get_TriggerMode (tagNATriggerMode *pMode)`
`HRESULT put_TriggerMode (tagNATriggerMode newMode)`

Interface `IChannel`

6-Nov-2007 Added new sweep mode

TriggerSignal Property - Superseded

Description **Note:** This command has been replaced by [Source](#) Property

Sets or returns the trigger source.

VB Syntax *app*.TriggerSignal = *value*

Variable [\(Type\)](#) - Description

app An Application (**object**)

value (enum NATriggerSignal) - Choose from:

0 - naTriggerInternal - free run

1 - naTriggerExternalPositive - a trigger signal is generated when a TTL high is sensed on the external trigger pin of the Aux IO connector

2 - naTriggerExternalNegative - a trigger signal is generated when a TTL low is sensed on the external trigger pin of the Aux IO connector.

3 - naTriggerManual - manual trigger source; use *app*.[ManualTrigger](#) to send a trigger signal.

4 - naTriggerExternalHigh - a trigger signal is generated when a TTL high is sensed on the external trigger pin of the Aux IO connector

5 - naTriggerExternalLow - a trigger signal is generated when a TTL low is sensed on the external trigger pin of the Aux IO connector

Return Type Long Integer

Default naTriggerInternal

Examples

```
app.TriggerSignal = naTriggerExternalPositive 'Write
trigsig = app.TriggerSignal 'Read
```

C++ Syntax HRESULT get_TriggerSignal(tagNATriggerSignal *pSignal)
 HRESULT put_TriggerSignal(tagNATriggerSignal signal)

Interface IApplication

TriggerType Property - Superseded

Description **Note:** This property has been replaced with [Scope](#) Property.

Sets or returns the trigger type which determines the scope of a trigger signal.

VB Syntax *app.TriggerType = value*

Variable [\(Type\)](#) - Description

app An Application (**object**)

value (**enum NATriggerType**) - Trigger type. Choose from:

0 - naGlobalTrigger - a trigger signal is applied to all triggerable channels

1 - naChannelTrigger - a trigger signal is applied to the current channel. The next trigger signal will be applied to the next channel; not necessarily channel 1-2-3-4.

Return Type Long Integer

Default naGlobalTrigger

Examples

```
app.TriggerType = naGlobalTrigger 'Write
trigtyp = app.TriggerType 'Read
```

C++ Syntax HRESULT get_TriggerType(tagNATriggerType *pTrigger)
HRESULT put_TriggerType(tagNATriggerType trigger)

Interface IApplication

Type (calstd) Property

Description	Sets and Returns the type of calibration standard.
VB Syntax	<i>calstd.Type</i> = <i>value</i>
Variable	(Type) - Description
<i>calstd</i>	A CalStandard (object). Use calKit. GetCalStandard to get a handle to the standard.
<i>value</i>	(enum NACalStandardType) -Choose from: 0 - naOpen 1 - naShort 2 - naLoad 3 - naThru
Return Type	Long Integer
Default	Not Applicable
Examples	<pre>calstd.Type = naOpen 'Write</pre> <pre>standardtype = calstd.Type 'Read</pre>
C++ Syntax	HRESULT get_Type(tagNACalStandardType *pVal) HRESULT put_Type(tagNACalStandardType newVal)
Interface	ICalStandard

Type (testset) Property

Description	Returns the testset model number.
VB Syntax	<i>tset.Type model</i>
Variable	(Type) - Description
<i>model</i>	(String) Variable to return the Test set model
<i>tset</i>	A TestsetControl object.
	Obtained from the ExternalTestsets collection.
Return Type	String
Default	None
Examples	<code>testset.type model</code> See External Testset Program
C++ Syntax	HRESULT get_Type(BSTR *pType);
Interface	ITestsetControl

TZImag Property

Description Sets and Returns the TZImag value (the Imaginary Terminal Impedance value) for the calibration standard. Only applicable when "[Type](#)" is set to **naArbitraryImpedance**.

To set the other resistance values, use [TZReal](#)

VB Syntax `calstd.TZImag = value`

Variable [\(Type\)](#) - Description

calstd A CalStandard (**object**). Use calKit.[GetCalStandard](#) to get a handle to the standard.

value (**single**) - Value for TZImag in Ohms

Return Type Single

Default Not Applicable

Examples `calstd.TZImag = 15 'Write the value of TZImag to 15 Ohms`

`imp0 = calstd.TZImag 'Read the value of TZImag`

C++ Syntax HRESULT get_TZImag(float *pVal);
HRESULT put_TZImag(float newVal);

Interface ICalStandard2

TZReal Property

Description Sets and Returns the TZReal value (the real Terminal Impedance value) for the calibration standard. Only applicable when "[Type](#)" is set to **naArbitraryImpedance**.

To set the other resistance values, use [TZImag](#)

VB Syntax `calstd.TZReal = value`

Variable [\(Type\)](#) - Description

calstd A CalStandard (**object**). Use calKit.[GetCalStandard](#) to get a handle to the standard.

value (**single**) - Value for TZReal in Ohms

**Return
Type** Single

Default Not Applicable

Examples `calstd.TZReal = 15 'Write the value of TZReal to 15 Ohms`

`imp0 = calstd.TZReal 'Read the value of TZReal`

C++ Syntax HRESULT get_TZReal(float *pVal);
HRESULT put_TZReal(float newVal);

Interface ICalStandard2

Read-only

UnusedChannelNumbers Property

Description Returns an array of channel numbers that are NOT in use. An unused channel has NO measurements subscribed to it.

VB Syntax *chanNumbers* = *chans*.UnusedChannelNumbers (*NumberOfChannels*)

Variable [\(Type\)](#) - Description

chanNumbers Variable array to store the returned channel numbers

chans A [Channel collection](#) (object)

NumberOfChannels (**Long Integer**) Number of channels that you are requesting.

Return Type One-dimensional array of long integers. The size of the array is specified by the *NumberOfChannels* parameter.

Default Not Applicable

Examples `chanNumbers = chans.UnusedChannelNumbers(5)`

C++ Syntax HRESULT get_UnusedChannelNumbers(long numberRequested, VARIANT* channelNumbers);

Interface IChannels2

USBPowerMeterCatalog Property

Description	Returns the ID string of power meters / sensors that are connected to the PNA USB. Use the list to select a power sensor for a source power cal.
VB Syntax	<i>list</i> = <i>pwrCal</i> . USBPowerMeterCatalog
Variable	(Type) - Description
<i>list</i>	(String) Variable to store the returned list of USB power meters.
<i>pwrCal</i>	(object) – A SourcePowerCalibrator (object)
Return Type	Comma-delimited strings. Two sensor strings are separated by a semicolon.
Default	Not Applicable
Examples	<pre>Set pwrCal = pna.SourcePowerCalibrator list = pwrCal.USBPowerMeterCatalog 'Read</pre>
C++ Syntax	HRESULT get_USBPowerMeterCatalog(BSTR *pUSBList);
Interface	ISourcePowerCalibrator6

Last Modified:

24-Feb-2009 Updated with semicolon
11-Jul-2007 MX New topic

UseCalWindow Property - **Obsolete**

Description Replaced with [Custom Cal Window](#) commands.

Turns Calibration window ON or OFF during a calibration.
[Learn more.](#)

VB Syntax *guidedCal.UseCalWindow = value*

Variable [\(Type\) - Description](#)

guidedCal [GuidedCalibration](#) (object)

value **(Boolean)**

True Show calibration window

False Hide calibration window

Return Type **Boolean**

Default True

Example `guided.UseCalWindow = True`

C++ Syntax HRESULT get_UseCalWindow(VARIANT_BOOL* val);
HRESULT put_UseCalWindow(VARIANT_BOOL newVal);

Interface IGuidedCalibration

Last Modified:

27-Jan-2012 Obsolete
12-Sep-2007 MX New topic

Read-only

UsedChannelNumbers Property

Description	Returns an array of channel numbers that are in use. A used channel has at least one measurement subscribed to it
VB Syntax	<i>chanNumbers</i> = <i>chans</i> .UsedChannelNumbers
Variable	(Type) - Description
<i>chanNumbers</i>	Variable array to store the returned channel numbers
<i>chans</i>	A Channel collection (object)
Return Type	One-dimensional array of long integers
Default	Not Applicable
Examples	<code>chanNumbers = chans.UsedChannelNumbers</code>
C++ Syntax	HRESULT get_UsedChannelNumbers(VARIANT* channelNumbers);
Interface	IChannels2

Read/Write

UseMultipleSensors Property

Description	Enable and disable the use of multiple power sensors during a guided calibration.
VB Syntax	<i>guidPwrSensors.UseMultipleSensors = value</i>
Variable	(Type) - Description
<i>guidPwrSensors</i>	GuidedCalibrationPowerSensors Collection
<i>value</i>	(Boolean)
	True Use multiple power sensors
	False Do NOT use multiple power sensors
Return Type	Boolean
Default	False
Examples	<pre>guidedPwrSensors.UseMultipleSensors = True</pre> <pre>value = guidedPwrSensors.UseMultipleSensors</pre>
C++ Syntax	<pre>HRESULT get_UseMultipleSensors(VARIANT_BOOL* val);</pre> <pre>HRESULT put_UseMultipleSensors(VARIANT_BOOL newVal);</pre>
Interface	IGuidedCalibrationPowerSensors

Last Modified:

8-Feb-2011 New topic

UsePowerLossSegments Property

Description	Specifies if subsequent power readings will use of the loss table. (PowerLossSegments).
VB Syntax	<i>pwrCal.UsePowerLossSegments = value</i>
Variable	(Type) - Description
<i>pwrCal</i>	A SourcePowerCalibrator (object)
	A PowerSensorAsReceiver (Object)
<i>value</i>	(boolean)
	False – Do not use loss table
	True – Use loss table
Return Type	Boolean
Default	False
Examples	<pre>pwrSens.UsePowerLossSegments = True 'Write lossTableState = pwrSens.UsePowerLossSegments 'Read</pre>
C++ Syntax	<pre>HRESULT put_UsePowerLossSegments(VARIANT_BOOL bState); HRESULT get_UsePowerLossSegments(VARIANT_BOOL *bState);</pre>
Interface	ISourcePowerCalibrator IPowerSensorAsReceiver

Last Modified:

25-Aug-2009 MX New topic

UsePowerSensorFrequencyLimits Property

Description	Specifies if subsequent calls to the AcquirePowerReadings method will observe frequency values of the MinimumFrequency and MaximumFrequency properties.
VB Syntax	<i>pwrCal</i> .UsePowerSensorFrequencyLimits = <i>value</i>
Variable	(Type) - Description
<i>pwrCal</i>	(object) – A SourcePowerCalibrator (object)
<i>value</i>	(boolean) - False – Do not use power sensor frequency limits. An acquisition will use just one power sensor for the entire sweep, regardless of frequency. True – Use power sensor frequency limits. A requested acquisition will only succeed for those frequency points which fall between the MinimumFrequency and MaximumFrequency values of that PowerSensor. An acquisition will pause in mid-sweep if the frequency is about to exceed the MaximumFrequency value. When the sweep is paused in this manner, a sensor connected to the other channel input of the power meter can be connected to the measurement port in place of the previous sensor, and then the sweep completed by another call to AcquirePowerReadings. However, the MaximumFrequency specified for the second sensor would need to be sufficient for the sweep to complete.
Return Type	Boolean
Default	False
Examples	<pre>Set powerCalibrator = pna.SourcePowerCalibrator powerCalibrator.UsePowerSensorFrequencyLimits = True 'Write FreqCheck = powerCalibrator.UsePowerSensorFrequencyLimits 'Read</pre>
C++ Syntax	HRESULT put_UsePowerSensorFrequencyLimits(VARIANT_BOOL bState); HRESULT get_UsePowerSensorFrequencyLimits(VARIANT_BOOL *bState);
Interface	ISourcePowerCalibrator

UserCalsetPrefix Property

Description Sets and returns the prefix to be used when saving User Cal Sets that result from the Cal All session. The Meas Class and channel number are appended to this prefix for each calibrated channel.

Use [GeneratedCalsets](#) to read the saved cal set names.

If a Cal Set prefix is NOT set, the cal data for each channel will be saved only to cal registers.

VB Syntax `calAll.UserCalsetPrefix = prefix`

Variable [\(Type\)](#) - Description

`calAll` A [CalibrateAllChannels](#) (object)

`prefix` (String) User CalSet prefix.

Return Type String

Default Not Applicable

Examples `calAll.UserCalsetPrefix = "MyCalSet" 'Set`

`csPrefix = calAll.UserCalsetPrefix 'Returns the CalSet prefix`

C++ Syntax HRESULT get_UserCalsetPrefix (BSTR calsetPrefix);

HRESULT put_UserCalsetPrefix (BSTR* calsetPrefix);

Interface ICalibrateAllChannels

Last modified:

4-Jan-2012 New topic

UserRange Property

Description Assigns the marker to the specified User Range. This restricts the marker's x-axis travel to the User Range span, specified with [Start](#) and [Stop](#) values.

- Each channel has **16** user ranges.
- Markers and trace statistics can be restricted to any user range.
- More than one marker can occupy a user range.
- User ranges can overlap. For example:
 - User range 1: 3 GHz to 5 GHz
 - User range 2: 4 GHz to 6 GHz

Note: User ranges are especially useful in restricting marker searches to specific areas of the measurement.

VB Syntax `mark.UserRange = value`

Variable [\(Type\)](#) - Description

mark A [Marker](#) (object)

value (**long integer**) - User Range. Choose any number between 0 and 16 (0=Full Span)

Return Type Long Integer

Default 0 - Full Span

Examples `mark.UserRange = 1 'Write`

`UseRnge = mark.UserRange 'Read`

C++ Syntax HRESULT get_UserRange(long *pRangeNumber)
HRESULT put_UserRange(long lRangeNumber)

Interface IMarker

UserRangeMax Property

Description **Note:** This property on the Channel Object is superseded by the same property on the Measurement Object.

Sets the stimulus stop value for the specified User Range.

VB Syntax *chan*.**UserRangeMax**(*domainType*,*Rnum*) = *value* - **Superseded**

meas.**UserRangeMax**(*Rnum*) = *value*

mark.**UserRangeMax**(*Rnum*) = *value*

Variable **(Type) - Description**

chan A [Channel](#) (object) - **Superseded**

meas A [Measurement](#) (object)

mark A [Marker](#) (object)

To assign a marker to a User Range, use the [UserRange Property](#).

domainType This argument is no longer required. The domain type is inferred by the measurement or marker.

(enum NADomainType) - Choose from:

0 - naDomainFrequency

1 - naDomainTime

2 - naDomainPower

Rnum (**long integer**) - User Range number. Choose any number between 1 and 16 (0=Full Span)

value (**double**) - Stop value. Choose any number within the full span of the channel.

Return Type Double

Default The current stimulus setting for the channel

Examples

```
mark.UserRangeMax(1) = 3e9 'Write
```

```
meas.UserRangeMax(1) = 3e9 'Write
```

```
UserRngeMax = mark.UserRangeMax(1) 'Read
```

```
UserRngeMax = meas.UserRangeMax(2) 'Read
```

C++ Syntax HRESULT put_UserRangeMax(long rangeNumber, double max Value)

HRESULT get_UserRangeMax(long rangeNumber, double *max Value)

Interface IMeasurement

IMarker

Last Modified:

3-May-2011 Updated for measurement object

UserRangeMin Property

Description Sets the stimulus start value for the specified User Range.

This property uses different arguments for the channel and marker objects.

VB Syntax *chan*.**UserRangeMin**(*domainType*,*range*) = *value*

or

mark.**UserRangeMin**(*range*) = *value*

Variable [\(Type\)](#) - Description

chan A Channel (**object**)

mark A [Marker](#) (**object**)

To assign a marker to a User Range, use the [UserRange Property](#).

domainType **Note:** The Marker object does not require the *domainType* argument.

(**enum NADomainType**) Type of sweep currently implemented on the channel - Choose from:

0 - naDomainFrequency

1 - naDomainTime

2 - naDomainPower

3 - naDomainPhase

range (**long**) - User Range number. Choose any number between **1** and **16** (0=Full Span)

value (**double**) - Start value. Choose any number within the full span of the analyzer

Return Type Double

Default The current stimulus setting for the channel

Examples

```
mark.UserRangeMin(1) = 3e9 'Write
chan.UserRangeMin(naDomainFrequency,1) = 3e9 'Write
```

```
UseRngeMin = mark.UserRangeMin 'Read
```

```
UseRngeMin = chan.UserRangeMin 'Read
```

C++ Syntax HRESULT put_UserRangeMin(tagNADomainType domain, long rangeNumber, double minValue)
HRESULT get_UserRangeMin(tagNADomainType domain, long rangeNumber, double *minValue)

Interface IChannel

Last Modified:

3-May-2011 Updated for phase

UserDescriptionOfPNA Property

Description Sets and reads a description of the PNA used to perform the User Characterization. This description is stored with the characterization in the ECal module.

Set this description before sending [Initialize](#) or the default (empty string) will be used.

VB Syntax *userChar*.UserDescriptionOfPNA = *value*

Variable [\(Type\) - Description](#)

userChar An [ECalUserCharacterizer](#) Object

value (String) Descriptive text, limited to 14 characters maximum.

Return Type String

Default ""(Empty String)

Examples

```
userChar.UserDescriptionOfPNA = "My PNA"
```

C++ Syntax HRESULT get_UserDescriptionOfPNA(BSTR *info);

HRESULT put_UserDescriptionOfPNA(BSTR info);

Interface IECalUserCharacterizer

Last Modified:

2-Nov-2008 New topic (8.33)

UserName Property

Description Sets and reads the description of the person and/or company who is producing the ECal user characterization. This description is stored with the characterization in the ECal module.

Set this description before sending [Initialize](#) or the default (empty string) will be used.

VB Syntax *userChar.UserName* = *value*

Variable [\(Type\)](#) - [Description](#)

userChar An [ECalUserCharacterizer](#) Object

value (String) Descriptive text, limited to 19 characters maximum.

Return Type String

Default "" (Empty String)

Examples

```
userChar.UserName = "John Doe, Acme Inc."
```

C++ Syntax HRESULT get_UserName(BSTR *name);

HRESULT put_UserName(BSTR name);

Interface IECalUserCharacterizer

Last Modified:

2-Nov-2008 New topic (8.33)

UserPresetEnable Property

Description 'Checks' and 'clears' the enable box on the User Preset dialog box. This only affects subsequent Presets from the front panel user interface.

Regardless of the state of the User Preset Enable checkbox, the [app.Preset](#) command will always preset the PNA to the factory preset settings, and [app.UserPreset](#) will always perform a User Preset.

VB Syntax `app.UserPresetEnable = state`

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

state (**boolean**) Front Panel User Preset State. Choose from:

False – User Preset OFF

True – User Preset ON

Return Type Boolean

Default False

Examples `app.UserPresetEnable = True 'Write`

`upreset = app.UserPresetEnable 'Read`

C++ Syntax HRESULT get_UserPresetEnable(VARIANT_BOOL *pVal)
HRESULT put_UserPresetEnable(VARIANT_BOOL newVal)

Interface IApplication6

Valid Property

Description	Returns a boolean value to indicate if the current equation on the measurement is valid. For equation processing to occur, the equation must be valid and ON.
VB Syntax	<i>IsValid</i> = <i>eq</i> . Valid
Variable	(Type) - Description
<i>good</i>	(Boolean) Variable to store the returned value.
	True (1) - equation is valid
	False (0) - equation is NOT valid
<i>eq</i>	Equation (object)
Return Type	Boolean
Default	Not Applicable
Examples	<code>IsValid = eq.Valid 'Read</code>
C++ Syntax	HRESULT get_Valid(Boolean *equation)
Interface	IEquation

Last Modified:

18-Jun-2007 New topic

ValidConnectorTypes Property

Description Returns a list of all connector types for which there are calibration kits. Looks for connector types in mechanical cal kits, within PNA disk memory, and within the attached ECal Module memory.

Here are the more common connector types:

W-band waveguide	Type B	1.00 mm female
V-band waveguide	Type A (50) female	1.00 mm male
U-band waveguide	Type A (50) male	1.85 mm male
R-band waveguide	Type F (75) female	1.85 mm female
Q-band waveguide	Type F (75) male	2.92 mm female
K-band waveguide	Type N (75) female	2.92 mm male
P-band waveguide	Type N (75) male	APC 2.4 female
X-band waveguide	Type N (50) female	APC 2.4 male
7-16 female	Type N (50) male	APC 3.5 female
7-16 male		APC 3.5 male
		APC 7

VB Syntax *value* = *obj*.ValidConnectorTypes

Variable [\(Type\) - Description](#)

value (Variant) List of connector types

obj Any of the following:

[ECalUserCharacterizer](#) (object)

[GuidedCalibration](#) (object)

[SMCType](#) (object)

[VMCType](#) (object)

Return Type Variant

Default Not Applicable

Examples `value = SMC.ValidConnectorTypes`

C++ Syntax `HRESULT get_ValidConnectorTypes(VARIANT* connectorTypes);`

Interface IGuidedCalibration

SMCType

VMCType

Last Modified:

23-Feb-2011 Added list

1-Sep-2009 Added ECal User and within disk memory (A.09.00)

Value Property

Description Write or read a value (setting) for the current element.

This command is used to set both RF and IF Path Configuration.

- See [RF Path Configuration \(elements, value\)](#)
- See [IF Path Configuration \(elements,value\)](#)

VB Syntax *pathElement*.**Value** = *value*

Variable [\(Type\)](#) - Description

pathElement A [PathElement](#) (object)

value **(String)** Value for the element. Use *pathElement.Values* to return a list of valid settings for this element.

Return Type String

Default Not Applicable

Examples See Examples:

- [IFPathConfiguration Setup](#)
- [RF PathConfiguration Example](#)

C++ Syntax HRESULT get_Value(BSTR* pValue);

HRESULT put_Value(BSTR value);

Interface IPathElement

Last Modified:

27-Mar-2012 Several edits

14-Dec-2006 MX New topic

Values Property

Description Returns an array of valid settings that can be used with the element object.

See a [list of configurable elements and settings](#) for various PNA models.

VB Syntax *values* = *pathElement*.**Values**

Variable [\(Type\)](#) - [Description](#)

values **(Variant array)** Variable to store the array of valid settings for the element.

pathElement A [PathElement](#) **(object)**

Return Type Variant array

Default Not Applicable

Examples `settings=pathElement.Values`

C++ Syntax HRESULT Values(VARIANT* pValues);

Interface IPathElement

Last Modified:

14-Dec-2006 MX New topic

VelocityFactor Property

Description Sets the velocity factor to be used with Electrical Delay, Port Extensions, and Time Domain marker distance calculations.

VB Syntax `app.VelocityFactor = value`

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

value (double) - Velocity factor. Choose a number between: **0** and **10**
(.66 polyethylene dielectric; .7 PTFE dielectric)

Return Type Double

Default 1

Examples `app.VelocityFactor = .66 'Write`

`RelVel = app.VelocityFactor 'Read`

C++ Syntax HRESULT get_VelocityFactor(double *pVal)
HRESULT put_VelocityFactor(double newVal)

Interface IApplication

View Property

Description Sets (or returns) the type of trace displayed on the screen.

VB Syntax `meas.View = value`

Variable [\(Type\)](#) - Description

meas A measurement (**object**)

value (**enum NAView**) - Type of trace. Choose from:

0 - naData

1 - naDataAndMemory

2 - naMemory

3 - naNoTrace

Note: The **naData** trace may reflect the result of a [TraceMath](#) operation.

Return Type NAView

Default naData

Examples `meas.View = naData 'Write`

`trceview = meas.View 'Read`

C++ Syntax HRESULT get_View(tagNAView* pView)
HRESULT put_View(tagNAView newView)

Interface IMeasurement

Visible Property

Description Makes the Network Analyzer application visible or not visible. In the Not Visible state, the analyzer cycle time for making measurements can be significantly faster because the display does not process data.

VB Syntax `app.Visible = state`

Variable [\(Type\)](#) - Description

app An [Application](#) (**object**)

state (**boolean**)

False - Network Analyzer application **NOT** visible

True - Network Analyzer application **IS** visible

Return Type Boolean

Default True

Examples

```
app.Visible = False 'Write
```

```
vis = app.Visible 'Read
```

C++ Syntax HRESULT get_Visible(VARIANT_BOOL * bVisible)
HRESULT put_Visible(VARIANT_BOOL bVisible)

Interface IApplication

WGCutoffFreq Property

Description Sets or returns the value of the waveguide cut off frequency.

VB Syntax *meas.WGCutoffFreq = value*

Variable [\(Type\)](#) - Description

meas A Measurement (**object**)

value (**double**) - Frequency in Hertz.

Return Type Double

Default Not Applicable

Examples `Print meas.WGCutoffFreq 'prints the value of the waveguide cut off frequency`

C++ Syntax `HRESULT get_WGCutoffFreq(double *pVal);`
`HRESULT put_WGCutoffFreq(double newVal);`

Interface IMeasurement2

WideBandDectectionState Property

Description Set and read the pulse mode detection method.

VB Syntax *pulseMeas.WideBandDectectionState = bool*

Variable [\(Type\)](#) - Description

pulseMeas A [PulseMeasurementControl](#) (object)

bool **False** - Narrowband mode.

True - Wideband mode

Return Type Boolean

Default Based on pulse width.

Examples `pulse.WideBandDectectionState = True 'Write`

`value = pulse.WideBandDectectionState 'Read`

C++ Syntax HRESULT get_WideBandDectectionState(VARIANT_BOOL *pVal);

HRESULT put_WideBandDectectionState(VARIANT_BOOL newVal);

Interface IPulseMeasurementControl

Last Modified:

11-Mar-2010 New topic

Width Property

Description Sets the pulse width - the amount of time that the pulse is ON.

VB Syntax *pulse*.**Width** (*n*) = *value*

Variable [\(Type\)](#) - Description

pulse A [PulseGenerator](#) (**object**)

n (**Integer**) Pulse generator number. Choose from 0 to 4.

0 is the generator that pulses the ADC.

value (**Double**) Pulse width in seconds. Choose a value from about 33ns to about 70 seconds.

Return Type Double

Default 1e-4 sec

Examples `pulse.Width = 1ms 'Write`

`value = pulse.Width 'Read`

C++ Syntax HRESULT get_Width (integer pulse, double* width);

HRESULT put_Width (integer pulse, double width);

Interface IPulseGenerator

Last Modified:

16-Feb-2012 Edit for Ext Pulse gens

2-Jan-2007 MX New topic

Read-only

WindowNumber Property

Description	Returns the window number. You might use this property to identify a particular window so that you can create a new Measurement in that window.
VB Syntax	<i>value</i> = <i>win</i> . WindowNumber
Variable	(Type) - Description
<i>win</i>	A NAWindow (object)
<i>value</i>	(long integer) - Variable to store the returned window number
Return Type	Long Integer
Default	Not Applicable
Examples	<code>value = app.ActiveNAWindow.WindowNumber</code>
C++ Syntax	HRESULT (long* windowNumber);
Interface	INAWindow

WindowState Property

Description Sets or returns the window setting of Maximized, Minimized, or Normal. To arrange all of the windows, use app.[ArrangeWindows](#).

VB Syntax *object*.WindowState = *value*

Variable [\(Type\)](#) - Description

object An [Application](#) (object) - main window

or

A NaWindow (**object**) - data windows

value (**enum NAWindowStates**) - The window state. Choose from:
0 - naMinimized - Minimizes the window to an Icon on the lower toolbar
1 - naMaximized - Maximizes the window
2 - naNormal - changes the window size to the user defined setting (between Max and Min).

Return Type Long Integer

Default naMaximized

Examples

```
app.WindowState = naMinimized 'changes the Network Analyzer application
window to an icon. -Write
win.WindowState = naNormal 'changes the window defined by the win object
variable to user defined settings. -Write
```

```
winststate = app.WindowState 'Read
```

C++ Syntax HRESULT get_WindowState(tagNAWindowStates *pVal)
 HRESULT put_WindowState(tagNAWindowStates newVal)

Interface INAWindow
 IApplication

XAxisAnnotation Property

Description Returns the X-Axis annotation of the specified tuning sweep.

VB Syntax *value* = *embedLO*.XAxisAnnotation (*n*)

Variable [\(Type\)](#) - Description

value **(String)** Variable to store the returned data.

embedLO An [EmbeddedLODiagnostic](#) (**object**)

n (Long) Tuning sweep number. Use [NumberOfSweeps](#) to find the number of sweeps taken.

Default Not Applicable

Examples `data= embedLO.XAxisAnnotation 3 'read`

C++ Syntax HRESULT XAxisAnnotation (long sweep, BSTR* annotation);

Interface IEmbeddedLODiagnostic

Last Modified:

13-Apr-2007 MX New topic

XAxisPointSpacing Property

Description Sets X-axis Point Spacing for the displaytraces measured with segment sweeps on the active channel.

VB Syntax *chan.XAxisPointSpacing = value*

Variable [\(Type\)](#) - Description

chan A Channel (**object**)

value (**Enum as naStates**) - Choose from:

0 - naOFF - Turns X-axis Point Spacing OFF

1 - naON - Turns X-axis Point Spacing ON

Return Type Enum

Default 0 - naOFF

Examples `chan.XAxisPointSpacing = naOFF 'Write`

`xspac = chan.XAxisPointSpacing 'Read`

C++ Syntax HRESULT get_XAxisPointSpacing (tagNAStates *pState);
HRESULT put_XAxisPointSpacing (tagNAStates newState);

Interface IChannel2

XAxisStart Property

Description Returns the X-Axis start value of the specified tuning sweep.

VB Syntax *value* = *embedLODiag*.XAxisStart (*n*)

Variable [\(Type\)](#) - Description

value **(Double)** Variable to store the returned data.

embedLODiag An [EmbeddedLODiagnostic](#) (**object**)

n (Long) Tuning sweep number. Use [NumberOfSweeps](#) to find the number of sweeps taken.

Default Not Applicable

Examples `data= embedLO.XAxisStart 3 'read`

C++ Syntax HRESULT XAxisStart (long sweep, double* start);

Interface IEmbeddedLODiagnostic

Last Modified:

13-Apr-2007 MX New topic

XAxisStop Property

Description Returns the X-Axis stop value of the specified tuning sweep.

VB Syntax *value* = *embedLODiag.XAxisStop* (*n*)

Variable [\(Type\)](#) - Description

value **(Double)** Variable to store the returned data.

embedLODiag An [EmbeddedLODiagnostic](#) (**object**)

n (Long) Tuning sweep number. Use [NumberOfSweeps](#) to find the number of sweeps taken.

Default Not Applicable

Examples `data= embedLO.XAxisStop 3 'read`

C++ Syntax HRESULT XAxisStop (long sweep, double* start);

Interface IEmbeddedLODiagnostic

Last Modified:

13-Apr-2007 MX New topic

YAxisAnnotation Property

Description Returns the Y-Axis annotation of the specified tuning sweep.

VB Syntax *value* = *embedLO*.YAxisAnnotation (*n*)

Variable [\(Type\)](#) - Description

value **(String)** Variable to store the returned data.

embedLO An [EmbeddedLODiagnostic](#) (**object**)

n (Long) Tuning sweep number. Use [NumberOfSweeps](#) to find the number of sweeps taken.

Default Not Applicable

Examples `data= embedLO.YAxisAnnotation 3 'read`

C++ Syntax HRESULT YAxisAnnotation (long sweep, BSTR* annotation);

Interface IEmbeddedLODiagnostic

Last Modified:

13-Apr-2007 MX New topic

YScale Property

Description Sets or returns the Y-axis Per-Division value of the active trace.

VB Syntax *trace*.YScale = *value*

Variable [\(Type\)](#) - Description

trace A Trace (**object**)

value (**double**) - Scale /division number. Units and range depend on the current data format.

Return Type Double

Default 10 (db)

Examples `trac.YScale = 5 'Write`

`yscl = trac.YScale 'Read`

C++ Syntax HRESULT get_YScale(double *pVal)
HRESULT put_YScale(double newVal)

Interface ITrace

Z0 Property

Description	Sets and Returns the characteristic impedance for the calibration standard.
VB Syntax	<i>calstd.Z0</i> = <i>value</i>
Variable	(Type) - Description
<i>calstd</i>	A CalStandard (object). Use calKit. GetCalStandard to get a handle to the standard.
<i>value</i>	(single) -Impedance in Ohms
Return Type	Single
Default	Not Applicable
Examples	<pre>calstd.Z0 = 50 'Write impedance = calstd.Z0 'Read</pre>
C++ Syntax	HRESULT get_Z0(float *pVal) HRESULT put_Z0(float newVal)
Interface	ICalStandard

Abort Method

Description	Ends the current measurement sweep on the channel.
VB Syntax	<i>chan</i> .Abort [<i>sync</i>]
Variable	(Type) - Description
<i>chan</i>	(object) - A Channel object
<i>sync</i>	(boolean) - wait (or not) for the analyzer to stop before processing subsequent commands. Optional argument; if unspecified, value is set to False. Choose from: True - synchronize - the analyzer will not process subsequent commands until the current measurement is aborted. False - continue processing commands immediately
Return Type	None
Default	None
Examples	<pre>chan.abort True chan.abort</pre>
C++ Syntax	HRESULT Abort(VARIANT_BOOL bSynchronize);
Interface	IChannel

AbortPowerAcquisition Method

Description	Aborts a source power cal acquisition sweep that is currently in progress.
VB Syntax	<i>powerCalibrator</i> . AbortPowerAcquisition
Variable	(Type) - Description
<i>powerCalibrator</i>	(object) - A SourcePowerCalibrator object
Return Type	None
Default	Not Applicable
Examples	<code>powerCalibrator.AbortPowerAcquisition</code>
C++ Syntax	HRESULT AbortPowerAcquisition();
Interface	ISourcePowerCalibrator

AcquireCalStandard Method - Superseded

Description **Note:** This command has been replaced by [AcquireCalStandard2 Method](#), which provides for acquisition of sliding load standards. All other functionality is identical.

VB Syntax *cal.AcquireCalStandard std[,index]*

Variable **(Type) - Description**

cal A Calibrator (**object**)

std (**enum NACalClass**) Standard to be measured. Choose from:

- 1 - naClassA
- 2 - naClassB
- 3 - naClassC
- 4 - naClassD
- 5 - naClassE
- 6 - naReferenceRatioLine
- 7 - naReferenceRatioThru

SOLT Standards

- 1 - naSOLT_Open
- 2 - naSOLT_Short
- 3 - naSOLT_Load
- 4 - naSOLT_Thru
- 5 - naSOLT_Isolation

TRL Standards

- 1 - naTRL_Reflection
- 2 - naTRL_Line_Reflection
- 3 - naTRL_Line_Tracking
- 4 - naTRL_Thru
- 5 - naTRL_Isolation

index (**long integer**) number of the standard. Optional argument - Used if there is more than one standard required to cover the necessary frequency range. If unspecified, value is set to 1.

Note The behavior has changed with PNA revisions as follows:

- Before 6.01: Accepted 0 and changed it to 1
- 6.01 to 6.04: Did NOT accept 0
- 6.04.11 and higher: Accepts 0 and changes it to 1

Return Type None

Default Not Applicable

Examples `Cal.AcquireCalStandard naSOLT_Thru 'Write`

C++ Syntax HRESULT AcquireCalStandard(tagNACalClass enumClass, short standardNumber)

Interface ICalibrator

Last modified:

10/05/06 Modified Index argument.

AcquireCalStandard2 Method

Description Measures the specified standard from the selected calibration kit. The calibration kit is selected using [app.CalKitType](#).

For 2-port calibration, it is also necessary to specify direction with [AcquisitionDirection](#).

To omit Isolation from a 2-port calibration, do not Acquire a cal standard for naSOLT_Isolation.

For using two sets of standards, see [Simultaneous2PortAcquisition Property](#).

Note: This command replaces [AcquireCalStandard](#). This command provides for the acquisition of a sliding load cal. All other functionality is identical.

VB Syntax `cal.AcquireCalStandard2 std[,index][,slide]`

Variable [\(Type\)](#) - Description

cal A [Calibrator](#) (object)

std (**enum NACalClass**) Standard to be measured. Choose from:

- 1 - naClassA
- 2 - naClassB
- 3 - naClassC
- 4 - naClassD
- 5 - naClassE
- 6 - naReferenceRatioLine
- 7 - naReferenceRatioThru

SOLT Standards

- 1 - naSOLT_Open
- 2 - naSOLT_Short
- 3 - naSOLT_Load
- 4 - naSOLT_Thru
- 5 - naSOLT_Isolation

TRL Standards

- 1 - naTRL_Reflection
- 2 - naTRL_Line_Reflection

- 3 - naTRL_Line_Tracking
- 4 - naTRL_Thru
- 5 - naTRL_Isolation

[index] (**long integer**) Number of the standard. Optional argument - Used if there is more than one standard required to cover the necessary frequency range. If unspecified, value is set to 1.

Note The behavior has changed with PNA revisions as follows:

- Before 6.01: Accepted 0 and changed it to 1
- 6.01 to 6.04: Did NOT accept 0
- 6.04.11 and higher: Accepts 0 and changes it to 1

[slide] (**enum** as NACalStandardSlidingState) Optional argument. State of the sliding load. The slide should be set a minimum of five times. Seven is the maximum that can be stored. Choose from:

0 - **naNotSlidingStd** - not using a sliding load - Default if not specified.

1 - **naSlideIsSet** - slide is set for acquisition

2 - **naSlideIsDone** - this next acquisition will be the last. Calculations will then be performed.

Return Type None

Default Not Applicable

Examples

```
Cal.AcquireCalStandard2 naSOLT_Thru
Cal.AcquireCalStandard2 naSOLT_Thru,2,naNotSlidingStd
'measures the second standard listed in the class of naSOLT_Thru'
```

C++ Syntax HRESULT AcquireCalStandard2(NACalClass enumClass, long standardPosition, NACalStandardSlidingState slidingStandardState)

Interface ICalibrator

Last modified:

10/05/06 Modified Index argument.

AcquireCalConfidenceCheckECALEX Method

Description This method replaces [AcquireCalConfidenceCheckECAL](#)

Transfers confidence data from the specified ECal module into the measurement's memory trace. The data is transferred to the specified S-parameter on the same channel as this Calibrator object.

The characterization within the ECal module that the confidence data will be read from is specified by [ECALCharacterizationEx](#). The default value is 0.

Note: A confidence check can NOT be performed remotely from User Characterizations that are stored on the PNA disk.

VB Syntax `cal.AcquireCalConfidenceCheckECALEX Sparam [,ecalModule]`

Variable [\(Type\) - Description](#)

cal A Calibrator (**object**)

Sparam (**String**) S-parameter to transfer confidence data to. This parameter must be present on the same channel as the calibrator object.

ecalModule (**Integer**) – Optional argument. ECal module.

Choose from modules **1** through **8**

Use [IsECALModuleFoundEx](#) to determine the number of modules connected to the PNA

Use [GetECALModuleInfoEx](#) to return the model and serial number of each module.

Return Type None

Default Not applicable

Examples `Cal.AcquireCalConfidenceCheckECALEX "S11", 2`

C++ Syntax `HRESULT AcquireCalConfidenceCheckECALEX(BSTR strParameter, long moduleNumber = 1);`

Interface ICalibrator4

Last Modified:

15-Jun-2010 Added Note

AcquirePowerReadingsEx Method

Description This command replaces [AcquirePowerReadings Method](#)

Initiates a source power cal acquisition.

VB Syntax `powerCalibrator.AcquirePowerReadingsEx calMethod, acqdevice [,sync]`

Variable [\(Type\) - Description](#)

powerCalibrator **(object)** - A [SourcePowerCalibrator](#) object

calMethod **(enum NASourcePowerCalMethod)** Selects the calibration method to be used for the source power cal acquisition.

0 – naPowerMeter Use power meter for all readings.

1 - naPowerMeterAndReceiver Power meter for the first iteration; then use the reference receiver for remaining readings if necessary.

2 - naReceiver Use PNA measurement receiver for all readings.

acqdevice **(String)** The specific acquisition device to be used. NOT case sensitive. Choose from:

If *calMethod* = **naPowerMeter** or **naPowerMeterAndReceiver**, choose from:

- "ASEN" -- Sensor on power meter channel A.
- "BSEN" -- Sensor on power meter channel B.
- To use the sensor that currently corresponds to the frequency of interest, use the value from the [PowerAcquisitionDevice](#) property.

If *calMethod* = **naReceiver**, choose from:

- The receiver names for your specific PNA using either physical receiver notation or [logical receiver notation](#). For example, "a1" or "A".
- Any configured PMAR device name. [Learn more about PMAR Devices. See PMAR commands](#)

[sync] **(boolean)** Optional argument. If not specified, value is set to False. Choose from:

True (1) – The method does not return until this acquisition has completed (the program calling this method is halted while waiting for the method to return).

False (0) – The method initiates an acquisition then returns immediately (while the acquisition still proceeds). The program calling this method can then perform other operations during the acquisition.

Return Type None

Default Not Applicable

Examples `powerCalibrator.AcquirePowerReadingsEx naPowerMeter, "asen", True`

`powerCalibrator.AcquirePowerReadingsEx naReceiver, "b2"`

`powerCalibrator.AcquirePowerReadingsEx naReceiver, "MyPMAR"`

C++ Syntax HRESULT AcquirePowerReadingsEx (tagNASourcePowerCalMethod enumCalMethod, BSTR bstrAcqDevice, VARIANT_BOOL bSync);

Interface ISourcePowerCalibrator4

Last modified:

10-Feb-2011 Added PMAR

9/12/06 MQ New command to accommodate receiver only SPC

AcquireStep Method

Description Acquire the measurement data for the specified step in the calibration process. For an ECal User characterization this measures the ECal module.

Note: Guided Cal allows you to measure standards in any order. [See an example.](#)

VB Syntax *obj.AcquireStep* (*n*)

Variable [\(Type\)](#) - Description

obj Any of the following:

[GuidedCalibration](#) (object)

[SMCType](#) (object)

[VMCType](#) (object)

[ECalUserCharacterizer](#) (object) - Currently, only ONE step is required to measure the ECal module.

n Step number in the calibration process.

Use [GenerateSteps](#) to determine the total number of steps.

Use [GetStepDescription](#) to read the description of each step.

Return Type Not Applicable

Default Not Applicable

Examples `VMC.AcquireStep (3)`

C++ Syntax `HRESULT put_AcquireStep(long step);`

Interface SMCType

VMCType

IGuidedCalibration

IECalUserCharacterizer

4-Nov-2008 Added ECalUserChar object

20-Jan-2007 Added any order note.

Activate Method

Description Makes an object the Active Object. When making a measurement active, the channel and window the measurement is contained in becomes the active channel and active window.

In order to change properties on any of the active objects, you must first have a "handle" to the active object using the **Set** command. For more information, See [Getting a Handle to an Object](#).

You do not have to make an object "Active" to set or read its properties remotely. But an object must be "Active" to change its values from the front panel.

VB Syntax *object*.Activate

Variable [\(Type\)](#) - Description

object [Measurement](#) (**object**)

or

[Marker](#) (**object**)

Return Type Not Applicable

Default Not Applicable

Examples
`meas.Activate`
`mark.Activate`

C++ Syntax HRESULT Activate()

Interface IMeasurement
IMarker

ActivateMarker Method

Description	Makes a marker the Active Marker. Use <code>meas.ActiveMarker</code> to read the number of the active marker.
VB Syntax	<code>meas.ActivateMarker(Mnum)</code>
Variable	(Type) - Description
<i>meas</i>	A Measurement (object)
<i>Mnum</i>	(long integer) - the number of the marker to make active. Choose any marker number from 1 to 9.
Return Type	None
Default	Not Applicable
Examples	<pre>meas.ActivateMarker(1) 'Write</pre>
C++ Syntax	<code>HRESULT ActivateMarker(long lMarkerNumber)</code>
Interface	IMeasurement
Remarks	Use ReferenceMarkerState to control the Reference marker.

ActivateWindow Method

Description Makes a window object the Active Window.

In order to change properties on any of the active objects, you must first have a "handle" to the active object using the **Set** command. For more information, See [Programming the Analyzer Object Model](#).

You do not have to make an object "Active" to set or read its properties remotely. But an object must be "Active" to change its values from the front panel.

VB Syntax `app.ActivateWindow n`

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

n (**long**) Number of the window to make active

Return Type Window Object

Default Not Applicable

Examples `app.ActivateWindow 4`

C++ Syntax HRESULT ActivateWindow(long WindowNumber)

Interface IApplication

[See the PNA Object Model](#)

Add (channels) Method

Description	Creates a channel and returns a handle to it. If the channel already exists, it returns the handle to the existing channel.
VB Syntax	<i>chans</i> .Add (<i>item</i>)
Variable	(Type) - Description
<i>chans</i>	A Channel collection (object)
<i>item</i>	(variant) - Channel number.
Return Type	Channel
Default	Not Applicable
Examples	<code>chans.Add 3 'Creates channel 3</code>
C++ Syntax	HRESULT Add(VARIANT numVal, IChannel** pChannel)
Interface	IChannels

Add (measurement) Method

Description Adds a Measurement to the collection.

Note: This command is supported ONLY in a [standard measurement](#) channel.

Measurements can be added to ALL measurement class types using [CreateCustomMeasurementEx Method](#).

VB Syntax `meas.Add channel,param,source[,window]`

meas A [Measurements](#) collection (**object**)

channel (**long**) - Channel number of the new measurement.

param (**string**) - New parameter. Case insensitive.

For S-parameters:

[Any S-parameter that can be measured by your PNA.](#)

Single-digit port numbers can be separated by "_" (underscore). For example: "S21" or "S2_1"

Double-digit port numbers MUST be separated by underscore. For example: "S10_1"

For Ratioed measurements:

Any two receivers in your PNA separated by "/". For example: "A/R1"

See the [block diagram](#) showing the receivers in YOUR PNA.

For Unratioed (absolute power) measurements:

Any receiver in the PNA. For example: "A"

See the [block diagram](#) showing the receivers in YOUR PNA

With PNA Rev 6.2, **Ratioed** and **Unratioed** measurements can also use **logical receiver notation** to refer to receivers. This notation makes it easy to refer to receivers with an [external test set](#) connected to the PNA. You do not need to know which physical receiver is used for each test port. [Learn more.](#)

For ADC measurements

Any ADC receiver in the PNA followed by a comma, then the source port.

For example: "AI1_2" indicates the Analog Input1 with source port of 2.

[Learn more about ADC receiver measurements.](#)

For Balanced S-parameter measurements:

"*topology:Sabxy*"

topology - Choose from:

- **sbal** - single-ended to balanced
- **ssb** - single-ended / single-ended to balanced
- **bbal** - balanced to balanced

Sabxy -

Where

a - device output (receive) mode

b - device input (source) mode

(choose from the following for both a and b:)

- **d** - differential
- **c** - common
- **s** - single ended

x - device output (receive) logical port number

y - device input (source) logical port number

For example:"**sbal:sdd42**"

[See an example program](#)

For [Imbalance](#) and [Common Mode Rejection](#) measurements:

"*topology:parameter*" Choose from:

Choose this:	To get this:	
	Topology	Parameter
"SBAL:IMBSB"	single-ended to balanced	imbalance
"SBAL:CMRRSB1"	single-ended to balanced	common mode rejection (Sds21/Scs21)
"SBAL:CMRRSB2"	single-ended to balanced	common mode rejection (Ssd12/Scs12)
"SSB:IMB1SSB"	single-ended / single-ended to balanced	imbalance 1
"SSB:IMB2SSB"	single-ended / single-ended to balanced	imbalance 2
"SSB:CMRRSSB1"	single-ended / single-ended to balanced	common mode rejection (Sds31/Scs31)
"SSB:CMRRSSB2"	single-ended / single-ended to balanced	common mode rejection (Sds32/Scs32)
"BBAL:IMB1BB"	balanced to balanced	imbalance 1
"BBAL:IMB2BB"	balanced to balanced	imbalance 2
"BBAL:CMRRBB"	balanced to balanced	common mode rejection (Sdd21/Scs21)

source **(long integer)** - Source port number; if unspecified, value is set to 1. Only used for non-s-parameter measurements; ignored if s-parameter.

window **(long integer)** - Optional argument. Window number of the new measurement. If unspecified, the S-Parameter will be created in the Active Window. Choose between 1 and the [maximum number of windows allowed on the PNA](#). If unspecified, the measurement will be created in the Active Window.

See also [Traces, Channels, and Windows on the PNA](#)

Return Type None

Default None

Examples `meass.Add 3,"A/R1",1,1 'Adds A/R1 measurement to channel 3 in window 1`

C++ Syntax HRESULT Add(long ChannelNum, BSTR strParameter, long srcPort,
VARIANT_BOOL bNewWindow)

Interface IMeasurements

Last modified:

13-Jan-2012 Added std cha note

9/12/06 MQ Added logical receiver notation and number of windows.

Add (NAWindows) Method

Description	Add a window to the display. Does not add a measurement. The window number must not already exist.
VB Syntax	<code>wins.Add [item]</code>
Variable	(Type) - Description
<i>wins</i>	A NAWindow collection (object)
<i>item</i>	(variant) - Window number. Choose between 1 and the maximum number of windows allowed on the PNA .
	See also Traces, Channels, and Windows on the PNA
Return Type	Object
Default	Not Applicable
Examples	<code>wins.Add 3 'Creates a window number 3</code>
C++ Syntax	HRESULT Add(long windowNumber)
Interface	INAWindows

Last modified:

4-Mar-2009 Removed optional argument
9/12/06 Modified for number of windows

Add (PowerLossSegment) Method

Description	Adds a PowerLossSegment to the PowerLossSegments collection. Also, Adds a PowerLossSegmentPMAR to the PowerLossSegmentsPMAR collection. To ensure predictable results, it is best to remove all segments before defining a new list of segments. For each segment in the collection, do a seg.Remove . Segments and values can also be added using the CharacterizeAdaptor Macro .
VB Syntax	<i>segs</i> .Add (<i>item</i> [<i>size</i>])
Variable	(Type) - Description
<i>segs</i>	A PowerLossSegments collection A PowerLossSegmentsPMAR collection.
<i>item</i>	(variant) - Number of the new segment. If it already exists, a new segment is inserted at the requested position.
<i>size</i>	(long integer) - Optional argument. The number of segments to add, starting with item. If unspecified, value is set to 1. Add up to 9999 segments.
Return Type	None
Default	Not Applicable
Examples	<pre>segs.Add 1, 4 'Adds segments 1,2,3 and 4</pre>
C++ Syntax	HRESULT Add(VARIANT index, long size);
Interface	IPowerLossSegments

Last Modified:

14-May-2012	Increase limit
25-Aug-2009	Added PMAR

Add (PowerSensorCalFactorSegment) Method

Description	Adds a PowerSensorCalFactorSegment to the CalFactorSegments collection. Also Adds a PowerSensorCalFactorSegmentPMAR to the CalFactorSegmentsPMAR collection. To ensure predictable results, it is best to remove all segments before defining a new list of segments. For each segment in the collection, do a seg.Remove .
VB Syntax	<i>segs</i> . Add (<i>item</i> [<i>size</i>])
Variable	(Type) - Description
<i>segs</i>	A CalFactorSegments (collection) or A CalFactorSegmentsPMAR (collection)
<i>item</i>	(variant) - Number of the new segment. If it already exists, a new segment is inserted at the requested position.
<i>size</i>	(long integer) - Optional argument. The number of segments to add, starting with item. If unspecified, value is set to 1.
Return Type	None
Default	Not Applicable
Examples	<pre>segs.Add 1, 4 'Adds segments 1,2,3 and 4</pre>
C++ Syntax	HRESULT Add(VARIANT index, long size);
Interface	ICalFactorSegments ICalFactorSegmentsPMAR

Last Modified:

25-Aug-2009 Added PMAR

Add (segment) Method

Description Adds segments to the Segments collection, but does not turn the segments ON.

VB Syntax *segs.Add* (*item*, [*size*])

segs A [segments](#) collection (**object**)

item (**variant**) Number of the new segment. If it already exists, a new segment is inserted at the requested position.

size (**long integer**) Optional argument. The number of segments to add, starting with *item*. If unspecified, value is set to 1.

Return Type None

Default None

Examples

```
Segs.Add 1, 4 'Adds segments 1,2,3,and 4. (does NOT automatically turn segments ON)
```

C++ Syntax HRESULT Add(VARIANT index, long size);

Interface ISegments

Remarks To ensure predictable results, it is best to remove all segments before defining a segment list. For each segment in the collection, do a [seg.Remove](#).

Add (External Device) Method

Description	Adds an external device to the system. This is the same as clicking the New button and editing the name on the Configure an External Device dialog. Upon creation, all settings on the new device are set to the defaults. The device is not active until set using Ext.Dev.Active .
VB Syntax	<i>extDevices</i> .Add name
Variable	(Type) - Description
<i>extDevices</i>	An ExternalDevices (collection)
<i>name</i>	(String) - Name of the new external device.
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>extDevices.Add 'MySource' 'Creates a new external device'</code>
C++ Syntax	HRESULT Add (BSTR name)
Interface	IExternalDevices

Last Modified:

31-Jul-2009 MX New topic

Add (GuidedPowerSensors) Method

Description Adds a power sensor name and item number to be used during a source power calibration. Use when multiple power sensors are to be used to calibrate the entire frequency span. The Name is used to recognize the sensor in the User Interface.

Item numbers in the [GuidedCalibrationPowerSensors](#) collection are used to refer to the power sensor remotely. Use the [Count Property](#) to return the number of power sensor items that are configured for use on the channel.

The port number to be calibrated is set using the [PerformPowerCalibration Property](#).

[Learn about using multiple power sensors](#)

VB Syntax *sensors.Add* (*name*)

Variable [\(Type\)](#) - Description

sensors A [GuidedCalibrationPowerSensors](#) (collection)

name **(String)** - Name of the power sensor to add. The power sensor must be already configured as a PMAR device using this name. [Learn how to remotely configure a PMAR device.](#)

Return Type Not Applicable

Default Not Applicable

Examples `sensors.Add "pmar2"`

[See Example program](#)

C++ Syntax HRESULT Add(BSTR name);

Interface IGuidedCalibrationPowerSensors

Last Modified:

8-Feb-2011 New topic

Add (Testset) Method

Description Adds a testset to the ExternalTestsets Collection and loads the configuration file.

VB Syntax `testsets.Add (model,address)`

Variable [\(Type\)](#) - Description

testsets An [ExternalTestsets](#) (collection)

model **(String)** Model of the testset to be added, NOT case-sensitive.

There is no COM command to read a list of currently-supported test sets. However, the following SCPI command can be used with the following format:

```
string = SCPIStringParser.Execute ("SENSe:MULTiplexer:CATalog? ")
```

address **(Integer)** Address of the testset to be added.

Return Type Not Applicable

Default Not Applicable

Examples

```
testsets.Add("Z5623AK66",12) ' add Z5623AK66 test at address 12 to testsets collection
```

[See an example program](#)

C++ Syntax HRESULT Add(BSTR typename, long address)

Interface [IExternalTestsets](#)

AddSegment Method

Description	Adds the specified number of segments to the scratch mixer at the index position. All segments are added with default settings.
VB Syntax	<i>conv</i> . AddSegment <i>index,count</i>
Variable	(Type) - Description
<i>conv</i>	A Converter Object
<i>index</i>	(Long integer) Position at which to add segments. Valid index range is between 1 and the current segment count +1. Using count +1 adds segments to the end of the segment table. Use SegmentCount Property to read the current count in the Applied Mixer .
<i>count</i>	(Long integer) Optional argument. Number of segments to add. If unspecified, 1 segment is added.
Return Type	Not Applicable
Default	Not Applicable
Examples	<pre>mxr.AddSegment 1,5 'Adds 5 segments beginning at the first position.</pre> See example program
C++ Syntax	HRESULT AddSegment(long index, long count);
Interface	IConverter5

Last Modified:

26-Oct-2010 New command (A.09.33)

AllowAllEvents Method

Description	Sets event filtering to monitor all events in the analyzer. This is the default setting when subscribing to events. This could slow the measurement speed of the analyzer significantly.
VB Syntax	<code>app.AllowAllEvents</code>
Variable	(Type) - Description <i>app</i> An Application (object)
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>app.AllowAllEvents</code>
C++ Syntax	HRESULT AllowAllEvents()
Interface	IApplication

AllowChannelToSweepDuringCalAcquisition Method

Description Specifies the channel to sweep during a Calibration.

When this command is sent, the *SwpChan* channel is 'flagged' to be swept during calibration. The flag is cleared when the channel is deleted, if the Measurement Class is changed, or if all measurements are deleted from the channel. If the same channel number is recreated, this command must be sent again to sweep the channel during a calibration. The flag is NOT saved with an instrument state.

A Preset or Instrument State Recall deletes the channel.

VB Syntax *calMgr.AllowChannelToSweepDuringCalAcquisition* (*CalChan*, *SwpChan*, *State*)

Variable [\(Type\)](#) - Description

calMgr **(object)** - A [CalManager](#) object

CalChan **(long)** - Channel to be calibrated.

SwpChan **(long)** - The channel to sweep when waiting to measure a standard.

This channel must already exist with at least one measurement in the channel. If this channel is in continuous sweep mode, it must have the same attenuator settings and path configuration (PNA-X only).

state **(Boolean)** - Channel sweep state. Choose from:

True - Sweep the channel during calibration.

False - Do NOT sweep the channel during calibration.

Return Type Not Applicable

Default Not Applicable

Example `calMgr.AllowChannelToSweepDuringCalAcquisition 2,1,True`

[See example using this command](#)

C++ Syntax HRESULT AllowChannelToSweepDuringCalAcquisition (long CalChannel, long SwpChannel, VARIANT_BOOL bVal);

Interface ICalManager5

23-Jan-2009 Added 'if measurements are deleted'.
8-Nov-2007 MX New topic

AllowEventCategory Method

Description Sets event filtering to monitor a category of event.

VB Syntax *app.AllowEventCategory, category, state*

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

category Category to monitor. Choose from list in [Working with the Analyzer's Events](#)

state **(boolean)**

True - monitor

False - do not monitor

Return Type Not Applicable

Default Not Applicable

Examples `app.AllowEventCategory`

C++ Syntax HRESULT AllowEventCategory(tagNAEventCategory category, VARIANT_BOOL bAllow)

Interface IApplication

AllowEventMessage Method

Description	Sets event filtering to monitor specific events.
VB Syntax	<i>app.AllowEventMessage event</i>
Variable	(Type) - Description
<i>app</i>	An Application (object)
<i>event</i>	Event to monitor. Refer to list in Working with the Analyzer's Events
<i>state</i>	(boolean) True - monitor False - do not monitor
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>app.AllowEventMessage</code>
C++ Syntax	HRESULT AllowEventMessage(tagNAEventID eventID, VARIANT_BOOL bAllow)
Interface	IApplication

AllowEventSeverity Method

Description	Sets event filtering to monitor levels of severity.
VB Syntax	<i>app.AllowEventSeverity severity,state</i>
Variable	(Type) - Description
<i>app</i>	An Application (object)
<i>severity</i>	(enum naEventSeverity) Choose from:naEventSeverityERROR naEventSeverityINFORMATIONAL naEventSeveritySUCCESS naEventSeverityWARNING
<i>state</i>	(boolean) True - monitor False - do not monitor
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>app.AllowEventSeverity</code>
C++ Syntax	HRESULT AllowEventSeverity(tagNAEventSeverity severity, VARIANT_BOOL bAllow)
Interface	IApplication

Write only

Apply Method

Description	Applies the mixer setup and turns the channel ON. (Performs the same function as the Apply button on the mixer setup dialog box .) Learn about the Scratch and Applied mixer properties
VB Syntax	<i>obj</i> . Apply
Variable	(Type) - Description
<i>obj</i>	A Mixer Interface pointer to the Measurement (object)
	Or
	A Converter Object
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>mxr.Apply</code>
C++ Syntax	HRESULT Apply()
Interface	IMixer3
	IConverter

Last Modified:

2-Feb-2009 Added converter object

ApplyDeltaMatchFromCalSet Method

Description Specifies a Cal Set as a source of delta match correction.

If 'GUID' is not supplied then the Global Delta Match Cal Set is assumed. An error is returned if the specified Cal Set does not meet the following Delta Match criteria. The Global Delta Match Cal can ALWAYS be applied.

- Must have been performed using ECal or as a guided mechanical cal (not Unguided).
- Must have the same start freq, stop freq, and number of points as the channel being calibrated.
- Must calibrate the ports that are required by the TRL or Unknown Thru Cal as indicated by [PortsNeedingDeltaMatch Property](#).

[Learn more about Delta Match calibration.](#)

VB Syntax *guided*.ApplyDeltaMatchFromCalSet [*GUID*]

Variable [\(Type\)](#) - Description

guided [GuidedCalibration](#) (object)

GUID Optional Argument. GUID of the Cal Set to use. If unspecified, the Global Delta Match Cal Set is used.

Return Type Not Applicable

Default Not Applicable

Examples

```
guided.ApplyDeltaMatchFromCalSet "{2B893E7A-971A-11d5-8D6C-00108334AE96}"
```

C++ Syntax HRESULT ApplyDeltaMatchFromCalSet(BSTR calsetGUID);

Interface IGuidedCalibration2

ApplyPowerCorrectionValuesEx Method

Description	<p>This command replaces ApplyPowerCorrectionValues Method.</p> <p>Applies the array of power correction values to the channel memory and turns correction ON. Perform after completing a source power cal acquisition sweep.</p> <p>This command does NOT save the correction values. To save correction values, save an instrument / calibration state (*.cst file) after performing a source power cal.</p> <p>Optionally, as part of the source power calibration, perform calibration of the reference receiver used in the power calibration. Learn more.</p>
VB Syntax	<code>powerCalibrator.ApplyPowerCorrectionValuesEX [rRec]</code>
Variable	(Type) - Description
<code>powerCalibrator</code>	(object) - A SourcePowerCalibrator object
<code>rRec</code>	(Enum as <code>NASourcePowerApplyCorrectionOption</code>) Optional argument. Choose from:
	<p>0 - naSourcePowerApplyCorrectionDefault Do NOT perform and save a calibration of the reference receiver. (Default if not specified).</p> <p>1 - naIncludeReferenceReceiverPowerCal Perform and save a calibration of the reference receiver. The Cal Set, which includes only the reference receiver cal, is saved to the destination specified by RemoteCalStoragePreference.</p>
Return Type	None
Default	Not Applicable
Examples	<pre>powerCalibrator.ApplyPowerCorrectionValuesEX</pre> <pre>powerCalibrator.ApplyPowerCorrectionValuesEX (naIncludeReferenceReceiverPowerCal)</pre>
C++ Syntax	<code>HRESULT ApplyPowerCorrectionValuesEx(enum NASourcePowerApplyCorrectionOption option);</code>
Interface	<code>ISourcePowerCalibrator5</code>

Last Modified:

23-Apr-2007 MX New topic

ApplySourcePowerCorrectionTo Method

Description Copies and applies an existing Source Power Calibration to another channel.

VB Syntax *chan*.**ApplySourcePowerCorrectionTo** (*fromPortNum*, *targetChan*, *targetPortNum*);

Variable [\(Type\)](#) - Description

chan A [Channel](#) (**object**)

fromPortNum (Long) Port number of the existing source power correction.

targetChan (Long) Channel number to which the source power correction will be copied.

targetPortNum (Long) Port number to which the source power correction will be applied.

Return Type Not Applicable

Default Not Applicable

Examples `chan.ApplySourcePowerCorrectionTo 1,2,1`

C++ Syntax HRESULT ApplySourcePowerCalibrationTo (long fromPortNumber, long otherChannelNumber, long portNumber);

Interface IChannel11

Last Modified:

20-Jul-2007 MX New topic

Write only

AssignSourceToRole Method

Description	Assigns a configured source to the specified role. Use chan.RoleDevice for non-converter channels.
VB Syntax	<i>conv</i> . AssignSourceToRole <i>role, source</i>
Variable	(Type) - Description
<i>conv</i>	A Converter Object
<i>role</i>	(String) Role to which the external source is assigned. Choose from: For IMDX and IMSX, choose from: "RF2" "LO1" "LO2" For all other converter applications, choose from: "LO1" "LO2"
<i>source</i>	(String) Source name from Source Configuration dialog .
Return Type	Not Applicable
Default	Not Applicable
Examples	<pre>conv.AssignSourceToRole "LO1", "LO1Name"</pre>
C++ Syntax	HRESULT AssignSourceToRole(BSTR roleID, BSTR deviceName);
Interface	IConverter

Last Modified:

3-Jan-2012	Remove superseded (thx SW)
3-May-2011	Superseded
22-Feb-2011	Edited for FCA2
2-Feb-2009	New topic

AutoOrient Method

Description Returns the ECal port that is connected to the specified PNA port. A calibration does not have to be in process.

VB Syntax *ecalPortNumber* = *ecal*.**AutoOrient** (*chanNum*, *pnaPort*, *ecalCharNum*)

Variable [\(Type\)](#) - Description

ecalPortNumber (Long) Variable to store the returned ECal port number that is connected to the specified PNA port number. The returned ECal port number is a 1-based number: 1 = Port A, 2 = Port B, 3 = Port C, 4 = Port D.

Zero (0) is returned when the auto-orientation routine is unable to resolve the orientation.

ecal A [ECalModule Object](#) (**object**)

chanNum (Long) Channel number that contains the frequency range that will be calibrated.

pnaPort (Long) PNA port number.

ecalCharNum (Long) User Characterization number that matches the physical adapters/fixtures that are on the ECal module. This aids in determining the orientation of the ECal module.

Choose from:

0 Factory characterization (no adapters - data that was stored in the ECal module by Agilent)

1 User characterization #1

2 User characterization #2

...and so forth up to:

12 User characterization #12

Return Type Long Integer

Default Not Applicable

Examples

```
Dim app As AgilentPNA835x.Application

Set app = CreateObject("AgilentPNA835x.Application",
<analyzerName>)
```

```
Dim pna

pna.Preset

Const chanNum = 1

pna.Channels(chanNum).StopFrequency = 20E9 ' for a 20 GHz ECal
mod

Const pnaPortNumber = 1

Const ecalCharacterizationNum = 0

Dim calMgr

Set calMgr = pna.GetCalManager

Dim ecalPortNumber ' The returned ECal port number is a 1-based
number

' (1 = Port A, 2 = Port B, etc)

ecalPortNumber = calMgr.ECalModules(1).AutoOrient(chanNum,
pnaPortNumber, ecalCharacterizationNum)

MsgBox "ECal port number attached to PNA port 1 = " &
ecalPortNumber
```

[C++ Syntax](#)

HRESULT AutoOrient(long channel, long pnaPortNumber, long
characterization, long *pECalPortNumber);

[Interface](#)

IECalModule

Last Modified:

11-Apr-2011 Edited userChar verbage
6-Mar-2009 MX New topic

AutoPortExtMeasure Method

Description	Measures either an OPEN or SHORT standard. When this command is sent, the PNA acquires the measurement with which to set automatic port extensions. Learn more about choosing which standard to measure.
VB Syntax	<i>fixture</i> . AutoPortExtMeasure <i>value</i>
Variable	(Type) - Description
<i>fixture</i>	A Fixturing (object)
<i>value</i>	(Enum as NAAutoPortExtMeasure)
	0 - naAPEM_OPEN - Measure OPEN
	1 - naAPEM_SHORT - Measure SHORT
Return Type	ENUM
Default	Not Applicable
Examples	<code>fixture.AutoPortExtMeasure naAPEM_OPEN</code>
C++ Syntax	HRESULT get_AutoPortExtMeasure(tagNAAutoPortExtMeasure *pVal);
Interface	IFixturing2

AutoPortExtReset Method

Description Clears old port extension delay and loss data in preparation for acquiring new data. Send this command prior to sending a new series of measurements using [AutoPortExtMeasure Method](#). If acquiring both OPEN and SHORT standards, do not send this command between those acquisitions.

VB Syntax *fixture*.AutoPortExtReset

Variable [\(Type\)](#) - Description

fixture A [Fixturing](#) (object)

Return Type Not Applicable

Default Not Applicable

Examples `fixture.AutoPortExtReset`

C++ Syntax HRESULT AutoPortExtReset();

Interface IFixturing2

Autoscale Method

Description	<p>Trace Object - Autoscales only the ONE trace on which Autoscale is being called.</p> <p>NAWindow Object - Scales ALL of the traces to fit in the same window. This is equivalent to "Autoscale All" from the front panel.</p> <p>Autoscale (both trace and window) behaves differently when scale coupling is enabled. How it behaves depends on the scale coupling method. Learn more.</p>
VB Syntax	<i>object</i> .Autoscale
Variable	(Type) - Description
	<i>object</i> Trace (object)
	or
	NAWindow (object)
Return Type	Not Applicable
Default	Not Applicable
Examples	<pre>Trac.Autoscale 'Autoscales the trace Win.Autoscale 'Autoscales all the traces in the window -Write</pre>
C++ Syntax	HRESULT AutoScale()
Interface	INAWindow ITrace

Last Modified:

15-Sep-2010 Added links to scale coupling

AveragingRestart Method

Description Clears and restarts averaging of the measurement data.

VB Syntax `chan.AveragingRestart`

Variable [\(Type\)](#) - Description

chan A [Channel](#) (**object**)

Return Type Not Applicable

Default Not Applicable

Examples `chan.AveragingRestart`

C++ Syntax HRESULT AveragingRestart()

Interface IChannel

BuildHybridKit Method

Description Use this method when you have different port connectors. This is a convenient way to combine two kits that match the connectors on your DUT.

VB Syntax `app.BuildHybridKit port1Kit,p1sex,port2Kit,p2sex,adapter,user kit`

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

port1Kit (enum **NACalKit**) - Specifies the two kits to be used to build the hybrid kit.

port2Kit Choose from:

naCalKit_85032F_N50
naCalKit_85033E_3_5
naCalKit_85032B_N50
naCalKit_85033D_3_5
naCalKit_85038A_7_16
naCalKit_85052C_3_5_TRL
naCalKit_User7
naCalKit_User8
naCalKit_User9
naCalKit_User10

p1sex (enum **NAPortSex**) - Specifies the sex of the connector at that port. Choose from:

p2sex **naMale**
naFemale
naDon'tCare

adapter (enum **NAAdapter**) - Choose from:

naUserkit - the electrical length of the adapter in the userKit specifications
naZeroLength - no adapter

userKit (enum **NACalKit**) - The Hybrid kit - Choose from the previous list of kits

Return Type Not Applicable

Default Not Applicable

Examples

```
app.BuildHybridKit
naCalKit_85033E_3_5,naMale,naCalKit_85038A_7_16
,naFemale,naUserkit,naCalKit_User8
```

C++ Syntax HRESULT BuildHybridKit/(tagNACalKit port1Kit, tagNAPortSex port1Sex, tagNACalKit port2Kit, tagNAPortSex port2Sex, tagNAAdapter adapter, tagNACalKit userKit)

Interface IApplication

CalculateErrorCoefficients Method

Description This method is the final call in a calibration process. It calculates error-correction terms, turns error-correction ON and saves the error-correction terms to the channel's Cal Register or a User Cal Set.

Do NOT use this command during an ECAL.

Note: The destination (Cal Register or User Cal Set) is determined by the setting of the [RemoteCalStoragePreference](#) property.

VB Syntax *cal*.**CalculateErrorCoefficients**

Variable [\(Type\)](#) - Description

cal [Calibrator](#) (**object**)

Return Type Not Applicable

Default Not Applicable

Examples `Cal.CalculateErrorCoefficients`

C++ Syntax HRESULT CalculateErrorCoefficients()

Interface ICalibrator

Last Modified:

16-Apr-2007 MX Added link to Remote...

Calculate Method

Description Calculates the Input or Output frequencies of the mixer setup, applies the mixer setup to the mixer object, and turns the channel ON.

Note: There is also a [Calculate Method](#) on the Converter Object

VB Syntax *obj.Calculate (port)*

Variable [\(Type\)](#) - Description

obj A [Mixer Interface](#) pointer to the [Measurement](#) (object)

port (enum as MixerCalculation) Port of the mixer for which to calculate start and stop frequencies. Choose from:

enum	1st or only stage requires:	In addition, 2nd stage requires:
0 mixCalculateINPUT	<ul style="list-style-type: none"> • Output Start and Stop frequencies • LO frequency • Output sideband (High or Low) 	<ul style="list-style-type: none"> • IF Start and Stop frequencies • 2nd LO frequency • IF sideband (High or Low)
1 mixCalculateINPUT AndOUTPUT (2 stage mixers ONLY)	NA	<ul style="list-style-type: none"> • IF Start and Stop frequencies • Both LO frequencies
2 mixCalculateOUTPUT	<ul style="list-style-type: none"> • Input Start and Stop frequencies • LO frequency • Output sideband (High or Low) 	<ul style="list-style-type: none"> • IF Start and Stop frequencies • 2nd LO frequency • IF sideband (High or Low)
3 mixCalculateLO1	<ul style="list-style-type: none"> • Input Start and Stop frequencies • Output frequency • Output sideband (High or Low) 	<ul style="list-style-type: none"> • IF Start and Stop frequencies • 2nd LO frequency • IF sideband (High or Low)
4 mixCalculateLO2	NA	<ul style="list-style-type: none"> • Input Start and stop

frequencies

- 1st LO start and stop frequencies
- Output frequency
- IF sideband(High or Low)
- Output sideband(High or Low)

Return Type Not Applicable

Default Not Applicable

Examples `obj.Calculate (mixCalculateOUTPUT)`

C++ Syntax HRESULT Calculate()

Interface IMixer

Last Modified:

2-Feb-2009 Added converter

Calculate Method

Description Calculates the Input or Output frequencies of the mixer setup, applies the mixer setup to the mixer object, and turns the channel ON.

Note: There is also a [Calculate Method](#) on the IMixer Interface.

VB Syntax *obj.Calculate (port)*

Variable [\(Type\) - Description](#)

obj A [Converter Object](#)

port (enum as ConverterCalculation) Port of the mixer for which to calculate start and stop frequencies. Choose from:

enum	1st or only stage requires:	In addition, 2nd stage requires:
0 naCalculateINPUT	<ul style="list-style-type: none"> • Output Start and Stop frequencies • LO frequency • Output sideband (High or Low) 	<ul style="list-style-type: none"> • IF Start and Stop frequencies • 2nd LO frequency • IF sideband (High or Low)
1 naCalculateINPUT AndOUTPUT (2 stage mixers ONLY)	NA	<ul style="list-style-type: none"> • IF Start and Stop frequencies • Both LO frequencies
2 naCalculateOUTPUT	<ul style="list-style-type: none"> • Input Start and Stop frequencies • LO frequency • Output sideband (High or Low) 	<ul style="list-style-type: none"> • IF Start and Stop frequencies • 2nd LO frequency • IF sideband (High or Low)
3 naCalculateLO1	<ul style="list-style-type: none"> • Input Start and Stop frequencies • Output frequency • Output sideband (High or Low) 	<ul style="list-style-type: none"> • IF Start and Stop frequencies • 2nd LO frequency • IF sideband (High or Low)
4 naCalculateLO2	NA	<ul style="list-style-type: none"> • Input Start and stop

frequencies

- 1st LO start and stop frequencies
- Output frequency
- IF sideband(High or Low)
- Output sideband(High or Low)

Return Type Not Applicable

Default Not Applicable

Examples `obj.Calculate (naCalculateOUTPUT)`

C++ Syntax HRESULT Calculate()

Interface IConverter

Last Modified:

18-Feb-2011 MX New topic

ChangeParameter Method

Description Changes the parameter of the measurement.

VB Syntax *meas.ChangeParameter(param,src)*

Variable [\(Type\)](#) - Description

meas A [Measurement](#) (object)

param (**string**) - New parameter. Case insensitive.

For [S-parameters](#) and [Applications](#) parameters:

Single-digit port numbers can be separated by "_" (underscore). For example: "S21" or "S2_1"

Double-digit port numbers MUST be separated by underscore. For example: "S10_1"

For Ratioed receiver measurements:

Any two receivers in your PNA separated by "/". For example: "A/R1"

See the [block diagram](#) showing the receivers in YOUR PNA.

For Unratioed (absolute power) measurements:

Any receiver in the PNA. For example: "A"

See the [block diagram](#) showing the receivers in YOUR PNA

With PNA Rev 6.2, **Ratioed** and **Unratioed** measurements can also use **logical receiver notation** to refer to receivers. This notation makes it easy to refer to receivers with an [external test set](#) connected to the PNA. You do not need to know which physical receiver is used for each test port. [Learn more.](#)

For ADC measurements

Any ADC receiver in the PNA.

For example: "AI1" indicates the Analog Input1.

[Learn more about ADC receiver measurements.](#)

For Balanced S-parameter measurements:

"topology:Sabxy"

topology - Choose from:

- **sbal** - single-ended to balanced
- **ssb** - single-ended / single-ended to balanced
- **bbal** - balanced to balanced

Sabxy -

Where

a - device output (receive) mode

b - device input (source) mode

(choose from the following for both a and b:)

- **d** - differential
- **c** - common
- **s** - single ended

x - device output (receive) logical port number

y - device input (source) logical port number

For example: "**sbal:sdd42**"

[See an example program](#)

For [Imbalance](#) and [Common Mode Rejection](#) measurements:

"topology:parameter" Choose from:

Choose this:	To get this:	
	Topology	Parameter
"SBAL:IMBSB"	single-ended to balanced	imbalance
"SBAL:CMRRSB1"	single-ended to balanced	common mode rejection (Sds21/Scs21)
"SBAL:CMRRSB2"	single-ended to balanced	common mode rejection (Ssd12/Scs12)
"SSB:IMB1SSB"	single-ended / single-ended to balanced	imbalance 1
"SSB:IMB2SSB"	single-ended / single-ended to balanced	imbalance 2
"SSB:CMRRSSB1"	single-ended / single-ended to balanced	common mode rejection (Sds31/Scs31)
"SSB:CMRRSSB2"	single-ended / single-ended to balanced	common mode rejection (Sds32/Scs32)
"BBAL:IMB1BB"	balanced to balanced	imbalance 1
"BBAL:IMB2BB"	balanced to balanced	imbalance 2
"BBAL:CMRRBB"	balanced to balanced	common mode rejection (Sdd21/Scs21)

src (long integer)

- Ignored if *param* is an S-Parameter
- Source port if *param* is a ratioed or unratioed receiver measurement (including ADC measurements).

Note: If the port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-x model, then you must use [chan.getPortNumber](#) to translate the string into a port

number. To learn more see [Remotely Specifying a Source Port](#).

Return Type Not Applicable

Default Not Applicable

Examples

```
meas.ChangeParameter "S11",2 '2 is ignored
```

```
meas.ChangeParameter "VC21",1 '1 is ignored
```

```
meas.ChangeParameter "A/R1",2 '2 is the source port
```

```
meas.ChangeParameter "a1/b1",1 '1 is the source port
```

```
meas.ChangeParameter "R1",2 '2 is the source port
```

```
'to change to a parameter with a string name
dim app
set app = CreateObject("Agilentpna835x.application")
dim capabilities
set capabilities = app.Capabilities
dim portnum
portnum = Capabilities.GetPortNumber("Src2 Out1")
app.activemeasurement.ChangeParameter "A",portnum
```

[C++ Syntax](#) HRESULT ChangeParameter(BSTR parameter, long lPort)

Interface IMeasurement

Last Modified:

24-Apr-2008 Added example for string names
30-Apr-2007 Modified for ADC and src strings

CheckPower Method

Description	Measures power at a specified frequency. Use this method to test power level before and/or after applying a source power calibration.
VB Syntax	<i>pow</i> = <i>pwrCal</i> . CheckPower (<i>device</i> , <i>freq</i> [, <i>unit</i>])
Variable	(Type) - Description
<i>pow</i>	(double) Variable to store power value returned by this method.
<i>pwrCal</i>	A SourcePowerCalibrator (object)
<i>device</i>	(enum NAPowerAcquisitionDevice) The specific sensor on the power meter to be used for the acquisition. Choose from: 0 – naPowerSensor_A 1 – naPowerSensor_B To use the sensor that currently corresponds to the frequency of interest, use the value from the PowerAcquisitionDevice property.
<i>freq</i>	(double) Frequency (Hz) at which the sensor is to read the power.
<i>unit</i>	(enum NAPowerUnit) Optional argument. Choose from: naDBM – Returns the power in dBm.(default) naWATT – Returns the power in Watts.
Return Type	Double
Default	Not Applicable
Examples	<pre>watt = powerCalibrator.CheckPower(naPowerSensor_A, 1E9, naWATT)</pre>
C++ Syntax	HRESULT put_CheckPower(tagNAPowerAcquisitionDevice enumAcqDevice, double dFreq, tagNAPowerUnit enumPowerUnit, double *pdPower);
Interface	ISourcePowerCalibrator2

Clear Method

Description Clears the current diagnostic information.

VB Syntax *embedLO*.Clear

Variable [\(Type\)](#) - Description

*embedLO*Diag An [EmbeddedLODiagnostic](#) (**object**)

Default Not Applicable

Examples `embedLO.Clear 'write`

C++ Syntax HRESULT Clear();

Interface IEmbeddedLODiagnostic

Last Modified:

12-Apr-2007 MX New topic

Clear Method

Description Clears the FIFO data buffer.

VB Syntax *fifo*.Clear

Variable [\(Type\)](#) - Description

fifo A [FIFO](#) (object)

Default Not Applicable

Examples `fifo.Clear 'write`

C++ Syntax HRESULT Clear();

Interface IFIFO

Last Modified:

3-Nov-2008 MX New topic

CloseCalSet Method **Superseded**

Description	<p>This command is no longer necessary. The CalSet.get... and put... commands that required this command have been replaced,</p> <p>Closes read/write access to the Cal Set.</p> <p>See OpenCalSet for an explanation of gaining access to the Cal Set.</p> <p>When you are finished reading and writing data from or to the Cal Set, close the Cal Set. Subsequent read/writes will require a new OpenCal Set call.</p> <p>Reading and writing Cal Set data is performed with the PutStandard, GetStandard, PutErrorTerm, GetErrorTerm method calls. These methods are provided by the ICal Set and ICalData2 interfaces.</p>
VB Syntax	<i>CalSet</i> .CloseCalSet
Variable	(Type) - Description
<i>CalSet</i>	(object) - A Cal Set object
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>CSet.CloseCalSet</code>
C++ Syntax	HRESULT CloseCalSet
Interface	ICalSet

ComputeErrorTerms Method

Description Computes error terms for the caltype specified by a preceding OpenCal Set call.

The Cal Set must first be opened using [OpenCalSet](#). If this call has not been made, the following error is issued:

E_NA_Cal Set_ACCESS_DENIED

The standards data required for the CalType must be available in the Cal Set or this error will be returned: E_NA_STANDARD_NOT_FOUND.

Note: Error term computation requires data for the actual calibration kit standards from the current kit definition. ComputeErrorTerms assumes that the standards were acquired using only one standard per class.

VB Syntax *CalSet*.**ComputeErrorTerms**

Variable [\(Type\)](#) - Description

CalSet **(object)** - A [Cal Set](#) object

Return Type Not Applicable

Default Not Applicable

Examples `CalSet.ComputeErrorTerms`

C++ Syntax HRESULT ComputeErrorTerms()

Interface ICalSet

ConfigEnhancedNB2 Method

Description **Note:** This command replaces [ConfigEnhancedNB Method](#).

This subroutine determines, then returns, the proper configuration for pulsed measurements on the **PNA-X ONLY** using the spectral nulling technique. The configuration returned needs to be sent to the PNA and any other related external equipment.

The routine will take a desired Pulse Repetition Frequency (PRF) and measurement IFBW and return a possibly modified PRF and IFBW for proper pulsed operation on the PNA.

Note: If an error is returned suggesting that nulling has not been found, add a small offset to the PRF (for example, 2.1 MHz instead of 2 MHz) or set *Fixed PRF* to **False**.

VB Syntax *Pulsed*.**ConfigEnhancedNB2** (*PRF*, *BW*, *PhysicalIF*, *NCO*, *ClockFreq*, *Stage1TapArray*, *Stage2TapArray*, *Stage3TapArray*, *FixedPRF*, *GateDelay*, *GateWidth*, *SWGateDelay*, *SWGateWidth*, *SWGateRamp*)

Variable [\(Type\)](#) - **Description**

Pulsed **(interface)** An interface to the agilentpnapulsed.dll application interface.

PRF **(Double)** The Pulse Repetition Frequency.

[out] The pulse repetition frequency that has been optimized for use with the PNA. NOTE: This value may be different from the value requested.

[in] The desired pulse repetition frequency.

BW **(Long)** The PNA IF Bandwidth.

[out] The PNA IF bandwidth that has been optimized for use with the PNA. NOTE: This value may be different from the value requested. Zero (0) is returned if no solution is found for the specified *PRF* and *BW*.

[in] The desired PNA IF bandwidth.

PhysicalIF **(Double)**

[out] Returns physical intermediate frequency.

<i>NCO</i>	(Double)	
		[out] Returns numeric controlled oscillator frequency.
<i>ClockFreq</i>	(Double)	
		[out] Returns the clock frequency (in Hz) of the PNA-X.
<i>Stage1TapArray</i>	(Long array)	
		[out] Returns the stage 1 filter coefficients
<i>Stage2TapArray</i>	(Long array)	
		[out] Returns the stage 2 filter coefficients
<i>Stage3TapArray</i>	(Long array)	
		[out] Returns the stage 3 filter coefficients
<i>FixedPRF</i>	(Boolean)	
		[in]
		<ul style="list-style-type: none"> • 1 (True) Signals the .DLL routine to NOT adjust the PRF value; rather adjust ONLY the IF Bandwidth. This is the default setting. • 0 (False) Adjust both the PRF and IF Bandwidth values as necessary.
<i>GateDelay</i>	(Double)	
		[in] Highest delay value in seconds used in any of the receiver gates.
<i>GateWidth</i>	(Double)	
		[in] Widest pulse width value in seconds used in any of the receiver gates.
<i>SWGateDelay</i>	(Double)	
		[out] Returns the SW gate delay in seconds.
<i>SWGateWidth</i>	(Double)	
		[out] Returns the SW Gate width in seconds.
<i>SWGateRamp</i>	(Long)	
		[out] Returns the SW Gate ramp
Return Type		Not Applicable
Default		Not Applicable

Example [See an example using this command.](#)

C++ Syntax HRESULT **ConfigEnhancedNB2**(double *pPRF, long *pBW, double *pIF, double *pNCO, double *clock, double *pStg1, double *pStg2, double *pStg3, VARIANT_BOOL fixPRF, double gateDelay, double gateWidth, double *SWgateDelay, double *SWgateWidth, long *SWgateRamp)

Interface AgilentPNAPulsed.Application

Last Modified:

28-Mar-2007 MX New topic

ConfigEnhancedNBIFAtten Method

Description	Sets PNA-X receivers to auto gain setting.
VB Syntax	<i>Pulsed</i> . ConfigEnhancedNBIFAtten (<i>PRF</i> , <i>RxWidth</i> , <i>IFAtten</i>)
Variable	(Type) - Description
<i>Pulsed</i>	(interface) An interface to the agilentpnapulsed.dll application interface.
<i>PRF</i>	(Double) [in] The Pulse Repetition Frequency.
<i>RxWidth</i>	(Double) [in] Receiver gate width.
<i>IFAtten</i>	(Long Integer) [out] IF attenuation value.
Return Type	Not Applicable
Default	Not Applicable
Example	See an example using this command.
C++ Syntax	HRESULT ConfigEnhancedNBIFAtten (double *pPRF, double *pWidth, long *pIF)
Interface	AgilentPNAPulsed.Application

Last Modified:

18-Jun-2007 MX New topic

ConfigNarrowBand3 Method

Description **Note:** This method replaces ConfigNarrowBand2 Method. The BW argument now returns 0 if no solution is found for the specified PRF and BW. In addition, adjustments were made to the filter finder algorithm

This subroutine determines, then returns, the proper configuration for pulsed measurements on the PNA using the spectral nulling technique. The configuration returned needs to be sent to the PNA and any other related external equipment such as pulse generators. The routine will take a desired Pulse Repetition Frequency (PRF) and measurement IFBW and return a possibly modified PRF and IFBW for proper pulsed operation on the PNA.

The routine will also return the Sample Rate, Number of Taps, and Offset that must be sent to the PNA to configure it in pulsed mode using the spectral nulling technique.

Although the example below uses COM programming to communicate with the PNA, these commands can be replaced with SCPI equivalents.

Note: The pulsed application may set the offset frequency (option 080) of the PNA to some value other than zero (the default value). If the stop frequency is set to the maximum of the PNA model, then an error message may appear on the PNA stating that the response frequency has exceeded the maximum allowed frequency. To fix this, set the stop frequency to a value that is at least 2 KHz less than the maximum allowed.

For example, if you have a 20 GHz PNA, and the stop frequency is set to 20 GHz, and the error message appears, then set the stop frequency to 19.999998 GHz

VB Syntax *Pulsed*.**ConfigNarrowBand** (*PRF*, *NumTaps*, *BW*, *OffSet*, *SampleRate*, *Precision*, *FixedPRF*, *PG81110*)

Variable [\(Type\)](#) - Description

Pulsed **(interface)** An interface to the agilentpnapulsed.dll application interface.

PRF **(Double)** The Pulse Repetition Frequency.

[out] The pulse repetition frequency that has been optimized for use with the PNA. NOTE: This value may be different from the value requested.

[in] The desired pulse repetition frequency.

NumTaps **(Long)** The number of taps to send to the PNA for pulsed operation.

BW **(Long)** The PNA IF Bandwidth.

[out] The PNA IF bandwidth that has been optimized for use with the PNA.
NOTE: This value may be different from the value requested. Zero (0) is returned if no solution is found for the specified *PRF* and *BW*.

[in] The desired PNA IF bandwidth.

Offset **(Double)** The offset value to send to the PNA for pulsed operation. The offset value is used to adjust the PNA for the two different possible sample rates that may be returned.

SampleRate **(Double)**

[out] The sample rate to send to the PNA for pulsed operation.

[in] Passing a value of 6.2 us will make sure that the offset frequency is not shifted and therefore could be used with converter measurements. Otherwise enter 0.

Precision **(Double)** The precision variables sets the precision that will be used to decrement the PRF when running the configuration routines. This variable can be set to the precision required by the external pulse generators so that the configuration routine will not return a PRF that is not within the precision limits of the pulse generators.

FixedPRF **(Boolean)**

1 (True) Signals the .DLL routine to NOT adjust the PRF value; rather adjust ONLY the IF Bandwidth. This is the default setting.

0 (False) Adjust both the PRF and IF Bandwidth values as necessary.

PG81110 **(Boolean)**

1 (True) You are using an Agilent 81110 as the pulse generator. This allows increased accuracy in adjustments for offset and PRF.

0 (False) Not using an Agilent 81110.

Return Type Not Applicable

Default Not Applicable

Example Removed example from help file.

C++ Syntax HRESULT ConfigNarrowBand(double *pPRF, long *pNumTaps, long *pBW, double *pOffset, double *pSampleRate, int Precision)

Interface AgilentPNAPulsed.Application

Last Modified:

21-Sep-2007 Modified sample rate and

ConfigurationFile Method

Description	Recalls an Interface Control file from the hard drive into the analyzer.
VB Syntax	<i>IntControl</i> . ConfigurationFile (<i>filename</i>)
Variable	(Type) - Description
<i>IntControl</i>	An InterfaceControl (object)
<i>filename</i>	(string) - Full path, file name, and extension (.xml) of the file to recall. Files are typically stored in "c:\users\public\network analyzer\documents"
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>IntControl.ConfigurationFile ("c:\users\public\network analyzer\documents/MySettings.xml ")</code>
C++ Syntax	HRESULT ConfigurationFile(BSTR bstrFile)
Interface	IInterfaceControl

Configurations Property

Description	Returns an array of stored configuration names that can be used with DeleteConfiguration Method and LoadConfiguration Method
VB Syntax	<i>names</i> = <i>pathMgr</i> . Configurations
Variable	(Type) - Description
<i>names</i>	(Variant array) Variable to store the returned configuration names.
<i>pathMgr</i>	PathConfigurationManager (object)
Return Type	Variant array
Default	Not Applicable
Examples	<pre>names = path.Configurations</pre>
C++ Syntax	HRESULT get_Configurations (VARIANT* configurations);
Interface	IPathConfigurationManager

Last Modified:

14-Dec-2006 MX New topic

Configure Method

Description Restarts as an "N-port" PNA using the specified multiport test set.

[See other commands to configure multiport test sets.](#)

VB Syntax `app.Configure (model, address)`

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

model **String** - Model of the test set with which to restart.

Use "**Native**" to restart without a test set.

To see a list of supported test sets, use

address **Integer** - GPIB Address of the test set. Use 0 for native restart.

Return Type Not Applicable

Default Not Applicable

Examples [See an example using this command.](#)

```
app.Configure ("N44xx",18)
```

C++ Syntax HRESULT Configure(BSTR model, long address);

Interface IApplication9

Continuous Method

Description The channel continuously responds to trigger signals.

Note: This command does **NOT** change [TriggerSignal](#) to Continuous.

VB Syntax `chan.Continuous`

Variable [\(Type\)](#) - Description

chan A [Channel](#) (object)

Return Type Not Applicable

Default Not Applicable

Examples `chan.Continuous`

C++ Syntax HRESULT Continuous()

Interface IChannel

Copy Method

Description Creates a new Cal Set and copies the current Cal Set data into it. Therefore, you now have a clone Cal Set with a different ID. Use this command to manipulate data on a Cal Set without corrupting the original cal data.

VB Syntax *CalSet.Copy*

Variable [\(Type\)](#) - Description

CalSet **(object)** - A [Cal Set](#) object

Return Type Not Applicable

Default Not Applicable

Examples

```
Dim mgr As CalManager
Dim ocalset As CalSet
Dim newcalset As CalSet
Set mgr = pna.GetCalManager
'Create a new (empty) Cal Set.
Set ocalset = mgr.CreateCalSet(1)
ocalset.Description = "original calset"
pna.Channel(1).SelectCalSet ocalset.GetGUID, True

'Launch the cal wizard and allow the user to perform the
calibration.
If pna.LaunchCalWizard(False) Then
'If the Launch returns true then the calibration finished.
ocalset.Save

'Copy the Cal Set to the new one.
Set newcalset = ocalset.Copy
newcalset.Description = "copy of original calset"

Else
'If the cal doesn't finish, delete the old Cal Set
'so it isn't taking up unnecessary memory.
mgr.DeleteCalSet ocalset.GetGUID
End If
```

As a result, the programmer can manipulate the data in the new Cal Set and always revert back to the old Cal Set as needed.

C++ Syntax HRESULT Copy(ICalSet** pCalSet);

Interface ICalSet

CopyFrom Method

Description Copies the mechanical switch and attenuator settings from the specified channel to the calling channel.

To avoid potential conflicts, all port couplings in the calling channel will be turned OFF and all port attenuator settings will be set to manual before copying the switch or attenuator settings. The two channels CAN be of different measurement classes.

Use [CopyToChannel](#) to copy ALL settings from one channel to another.

VB Syntax *pathConfig*.**CopyFrom** (*chanNum*)

Variable [\(Type\)](#) - **Description**

pathConfig A [PathConfiguration](#) (**object**)

chanNum (**long integer**) Channel number to copy to the calling channel.

Return Type None

Default Not Applicable

Examples

```
Dim chan,pathConfig
Set chan = app.ActiveChannel
Set pathConfig = chan.PathConfiguration
pathConfig.CopyFrom 2
```

C++ Syntax HRESULT CopyFrom(long chanNum);

Interface IPathConfiguration2

Last modified:

3-May-2011 New topic

CopyToChannel Method

Description	Copies ALL settings from this channel to the specified channel. Use CopyFrom to copy ONLY the mechanical switch and attenuator settings.
VB Syntax	<i>chan</i> .CopyToChannel(<i>lChanNum</i>)
Variable	(Type) - Description
<i>chan</i>	A Channel (object)
<i>lChanNum</i>	(long integer) – Number of the channel to become a copy of <chan>.
Return Type	None
Default	Not Applicable
Examples	<pre>Dim chan Set chan = PNAapp.ActiveChannel chan.CopyToChannel 2</pre>
C++ Syntax	HRESULT CopyToChannel(long lChanNum);
Interface	IChannel2

Last modified:

3-May-2011 Added copyFrom link

CreateS-Parameter Method

Description	This method creates a new S-Parameter measurement in an existing or new window.
VB Syntax	<code>app.CreateSParameter chan,recvr,source,[window]</code>
Variable	(Type) - Description
<i>app</i>	Application (object)
<i>chan</i>	(long integer) - Channel number of the new measurement
<i>recvr</i>	(long integer) - Port number of the receiver (1 or 2)
<i>source</i>	(long integer) - Port number of the source (1 or 2)
<i>window</i>	(long integer) - Optional argument. Window number of the new measurement. Choose 1 to 4 . If unspecified, the S-Parameter will be created in the Active Window.
Return Type	Not Applicable
Default	Not Applicable
Examples	<pre>app.CreateSParameter 1,2,1,1 'Creates a new S21 measurement in channel 1 and New window(1) app.CreateSParameter 1,2,1 'Creates a new S21 measurement in channel 1 and in the active window</pre>
C++ Syntax	HRESULT CreateSParameter(long ChannelNum, long RcvPort, long SrcPort, long windowNumber)
Interface	IApplication

Last Modified:

12-Mar-2014 No longer superseded. Use instead of [Create SParameterEX Method](#)

CreateCalSet Method

Description Creates a new Cal Set.

The new cal set is initialized with the stimulus settings from the channel whose number is passed as the argument to this method. Stimulus settings include frequency, bandwidth, number of points, and so forth.

Use this method when you want to manually upload data to the Cal Set using the returned ICal Set interface handle..

The channel number does not restrict the usage of this Cal Set on any other channel. It simply provides a link to the originating channel so that the stimulus values can be stored in the Cal Set.

Note: Be sure to SAVE the CalSet you are creating. Use [ICalSet::Save](#).

VB Syntax *calMgr*.CreateCalSet (*chan*)

Variable [\(Type\)](#) - Description

calMgr **(object)** - A [CalManager](#) object

chan **(long)** - channel number of the new Cal Set.

Return Type ICal Set Interface

Default Not Applicable

Example `calMgr.CreateCalSet 1`

C++ Syntax HRESULT CreateCalSet(long ChannelNumber, ICal Set** pCal Set);

Interface ICalManager

Write-only

CreateCustomCal Method

Description	Creates a custom cal object.
VB Syntax	<i>calMgr</i> . CreateCustomCal (<i>CalType</i>)
Variable	(Type) - Description
<i>calMgr</i>	Cal Manager (Object)
<i>CalType</i>	(String) Name of the calibration. Choose from: "VMC" or "VectorMixerCal.VMCTYPE" "SMC" or "ScalarMixerCal.SMCTYPE" See Also SMCTYPE Object VMCTYPE Object
Return Type	Not Applicable
Default	Not Applicable
Examples	<pre>Dim CalMgr As ICalManager2 Dim SMC As ISMCTYPE Set SMC = CreateCustomCal("SMC")</pre> See SMC and VMC examples using this command.
C++ Syntax	HRESULT CreateCustomCal(BSTR CustomCal)
Interface	ICalManager2

CreateCustomCalEx Method

Description Returns [IGuidedCalibration](#) for the specified channel.

With a handle to the IGuidedCalibration interface, you can query it for the following objects for properties appropriate to the calibration setup for the particular meastype (channel).

IGuidedCalibration interface is used to configure a calibration (specify connectors, cal kits, and so forth). It is also used to access any custom calibration properties required for unique application channels like Noise Figure or Gain Compression. To access these special properties, make this call on the IGuidedCalibration interface:

```
CustomInterface = IGuidedCalibration.CustomCalConfiguration\(\);
```

The interface returned by this call can be used to set and get the custom properties on the following application cal objects:

- [NoiseCal Object](#)
- [GainCompressionCal Object](#)
- [SweptIMDCal Object](#)

Note: Use [CreateCustomCal_Method](#) to create FCA calibration objects.

VB Syntax *calmgr*.**CreateCustomCalEx** (*chan*)

Variable [\(Type\)](#) - **Description**

calMgr [Cal Manager](#) (**Object**)

chan (**long integer**) Channel number in which to create the Cal object.

Return Type [IGuidedCalibration](#)

Default Not Applicable

Examples

```
Dim guidedcal
Set guidedcal = CalManager.CreateCustomCalEx(1)
```

See Also

- [Noise Figure example](#)
- [Gain Compression example](#)

C++ Syntax HRESULT CreateCustomCalEx(long channel, IDispatch** ppObject);

Interface ICalManager5

Last Modified:

22-Jul-2009 Added description text per SW

29-May-2007 MN New topic

CreateCustomMeasurementEx Method

Description Creates a new custom measurement or a new 'standard' S-Parameter measurement.

VB Syntax *app.CreateCustomMeasurementEx chanNum,MeasClass,MeasName [,window]*

Variable (Type) - Description

app **(object)** - An [Application](#) object

chanNum **(long)** -Channel number used by the new measurement; can exist or be a new channel.

MeasClass **(string)** - Measurement class of the new custom measurement object. The Measurement Class must be installed and registered on the PNA.

Choose from the following:

- "Standard"
- "Vector Mixer/Converter"
- "Scalar Mixer/Converter"
- "Gain Compression"
- "Gain Compression Converters"
- "Noise Figure Cold Source"
- "Noise Figure Converters"
- "Swept IMD"
- "IM Spectrum"
- "Swept IMD Converters"
- "IM Spectrum Converters"

MeasName **(variant)** Measurement names to create:

Meas Class	Measurement Name	Description
Vector Mixer/Converter	"S11"	Learn about VMC parameters
	"VC21"	
	"S22"	

<p>Scalar Mixer/Converter</p>	<p>"S11" Learn about SMC parameters</p> <p>"SC21"</p> <p>"SC12"</p> <p>"S22"</p> <p>"Ipwr"</p> <p>"RevIPwr"</p> <p>"Opwr"</p> <p>"RevOPwr"</p>
<p>Gain Compression</p> <p>Learn more</p> <p>Gain Compression Converters</p> <p>Learn more</p>	<p>GCA and GCX:</p> <p>"CompIn21" Input power at the compression point.</p> <p>"CompOut21" Output power at the compression point.</p> <p>"CompGain21" Gain at the compression point.</p> <p>"CompS11" Input Match at the compression point</p> <p>"RefS21" Linear Gain</p> <p>"DeltaGain21" CompGain21 -Linear Gain</p> <p>"S11", "S21", "S12", "S22" Standard S-parameters; measured at port 1 and port 2</p> <p>GCX - All Gain Compression parameters (except S21 and S12) plus the following:</p> <p>"S11"</p> <p>"SC21"</p> <p>"SC12"</p> <p>"S22"</p> <p>"Ipwr"</p> <p>"RevIPwr"</p> <p>"Opwr"</p> <p>"RevOPwr"</p> <p>Mixer parameters</p>

Noise Figure Cold Source

[Learn more](#)

Noise Figure Converters

[Learn more](#)

Noise Figure AND NFX:	
"NF"	Noise figure
"ENR"	Validate noise source measurements.
"T-Eff"	Effective noise temperature.
"DUTRNP" "DUTRNPI"	DUT noise power ratio. (Noise power expressed in Kelvin divided by 290).
"SYSRNP" "SYSRNPI"	System noise power ratio
"DUTNPD" "DUTNPDI"	DUT noise power density. (Noise power expressed in dBm/Hz).
"SYSNPD" "SYSNPDI"	System noise power density.
"OvrRng" (Opt 029 Only)	Indication that the noise receiver is being over powered.
"T-Rcvr" (Opt 029 Only)	Temperature reading (in Kelvin) of the noise receiver board.
Noise Figure ONLY - NOT NFX:	
"S11", "S21", "S12", "S22"	Standard S-parameters; measured with the port1 and port2 noise switches set for noise mode.
"A_1", "A_2" ...and so forth. "GammaOpt" "Rn" "NFMin"	Unratioed parameters; with notation: "receiver, source port" Optimum Complex Reflection Coefficient Noise Resistance Minimum noise figure that occurs at GammaOpt

	<p>NFX ONLY:</p> <p>"S11"</p> <p>"SC21"</p> <p>"SC12"</p> <p>"S22"</p> <p>"Ipwr" Mixer parameters</p> <p>"RevIPwr"</p> <p>"Opwr"</p> <p>"RevOPwr"</p> <hr/> <p>"ALO1", "BLO1" ...and so forth. Test port receiver at LO1 frequency</p> <hr/> <p>"R1_1", "B_2" ..and so forth. Unratioed parameters with notation: "receiver_source port"</p>
<p>Swept IMD</p> <p>Swept IMD Converters</p> <p>Learn more</p>	<p>There are over 150 possible Swept IMD parameters, too many to list here.</p> <p>Build the parameters with the Swept IMD Parameter dialog, then copy the parameter name to the remote command.</p> <p>The following are a few example parameters:</p> <p>"PwrMainLo" Absolute power of the Low tone at the DUT output.</p> <hr/> <p>"IM3" Power of the third product relative to the average power of the f1 and f2 tones measured at the DUT output.</p> <hr/> <p>"OIP3" Theoretical power level at which the third product will be the same power level as</p>

		the average of the main tones at the output of the DUT.
IM Spectrum Learn more	"Output"	View signals OUT of the DUT and into PNA port 2 (B receiver).
	"Input"	View signals IN to the DUT (R1 receiver).
	"Reflection"	View signals reflected off the DUT input and back into PNA port 1 (A receiver)
IMx Spectrum Converters Learn more	"Output"	View signals OUT of the DUT and into PNA port 2 (B receiver)

window **(long)** Optional argument. Number of the window the new custom measurement will be placed in. Choose between 1 and the [maximum number of windows allowed on the PNA](#).. If unspecified, the measurement is placed in the active window.

Return Type IMeasurement

Default Not Applicable

Examples

```
'To create a scalar mixer measurement in channel 2:
Dim MyMeas as Agilent835x.Measurement
Set MyMeas = app.CreateCustomMeasurementEx (2, "Scalar
Mixer/Converter", "SC21")
```

```
'To create a vector mixer measurement in channel 2:
Dim MyMeas as Agilent835x.Measurement
Set MyMeas = app.CreateCustomMeasurementEx (2, "Vector
Mixer/Converter", "VC21")
```

C++ Syntax HRESULT put_CreateCustomMeasurementEx (long ChannelNum, BSTR guid, VARIANT initData, long windowNumber, IMeasurement** ppMeasurement);

Interface IApplication3

14-Sep-2012	Fixed unratioed notation
27-Sep-2011	Added Opt 029 Only
2-Mar-2010	Fixed IM spectrum converters
29-Oct-2009	Added NFX
23-Feb-2009	Added IMD Converters and Noise Figure I
18-Sep-2008	Added IMD and IM Spectrum
8-Nov-2007	Updated for NF and GCA

CreateMeasurement Method

Description	Creates a new measurement.
VB Syntax	<i>app</i> .CreateMeasurement <i>chanNum,param,lPort[,window]</i>
Variable	(Type) - Description
<i>app</i>	Application (object)
<i>chanNum</i>	(long) - Channel number of the new measurement; can exist or be a new channel
<i>param</i>	(string) - New parameter. Case insensitive.

For S-parameters:

[Any S-parameter that can be measured by your PNA.](#)

Single-digit port numbers can be separated by "_" (underscore). For example: "S21" or "S2_1"

Double-digit port numbers MUST be separated by underscore. For example: "S10_1"

For Ratioed measurements:

Any two receivers in your PNA separated by "/". For example: "A/R1"

See the [block diagram](#) showing the receivers in YOUR PNA.

For Unratioed (absolute power) measurements:

Any receiver in the PNA. For example: "A"

See the [block diagram](#) showing the receivers in YOUR PNA

With PNA Rev 6.2, **Ratioed** and **Unratioed** measurements can also use **logical receiver notation** to refer to receivers. This notation makes it easy to refer to receivers with an [external test set](#) connected to the PNA. You do not need to know which physical receiver is used for each test port. [Learn more.](#)

For ADC measurements

Any ADC receiver in the PNA.

For example: "AI1" indicates the Analog Input1.

[Learn more about ADC receiver measurements.](#)

For Balanced S-parameter measurements:

"topology:Sabxy"

topology - Choose from:

- **sbal** - single-ended to balanced
- **ssb** - single-ended / single-ended to balanced
- **bbal** - balanced to balanced

Sabxy -

Where

a - device output (receive) mode

b - device input (source) mode

(choose from the following for both a and b:)

- **d** - differential
- **c** - common
- **s** - single ended

x - device output (receive) logical port number

y - device input (source) logical port number

For example: "**sbal:sdd42**"

[See an example program](#)

For [Imbalance](#) and [Common Mode Rejection](#) measurements:

"topology:parameter" Choose from:

Choose this:	To get this:	
	Topology	Parameter
"SBAL:IMBSB"	single-ended to balanced	imbalance
"SBAL:CMRRSB1"	single-ended to balanced	common mode rejection (Sds21/Scs21)
"SBAL:CMRRSB2"	single-ended to balanced	common mode rejection (Ssd12/Scs12)
"SSB:IMB1SSB"	single-ended / single-ended to balanced	imbalance 1
"SSB:IMB2SSB"	single-ended / single-ended to balanced	imbalance 2
"SSB:CMRRSSB1"	single-ended / single-ended to balanced	common mode rejection (Sds31/Scs31)
"SSB:CMRRSSB2"	single-ended / single-ended to balanced	common mode rejection (Sds32/Scs32)
"BBAL:IMB1BB"	balanced to balanced	imbalance 1
"BBAL:IMB2BB"	balanced to balanced	imbalance 2
"BBAL:CMRRBB"	balanced to balanced	common mode rejection (Sdd21/Scs21)

lPort (long)

- Ignored if *param* is an S-Parameter, balanced, imbalance, or CMRR parameter.
- Source port if *param* is ratioed or unratioed (including [ADC](#)) measurements.
- Use 0 for [N5264A](#).

Note: If the port is defined by a string name, such as an external source, a

balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-x model, then you must use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#).

window **(long)** Optional argument. Window number of the new measurement. Choose between 1 and the [maximum number of windows allowed on the PNA..](#) If unspecified, the measurement will be created in the Active Window.

See also [Traces, Channels, and Windows on the PNA](#)

Return Type Not Applicable

Default Not Applicable

Examples `app.CreateMeasurement(1, "A/R1", 1, 0)`

`app.CreateMeasurement(1, "a1/b1", 1, 0)`

`app.CreateMeasurement(1, "bba1:Sdd21", 1)`

`app.CreateMeasurement(1, "AI2", 2)`

`app.CreateMeasurement(1, "R1", 0) ' for N5264A`

C++ Syntax HRESULT CreateMeasurement(long ChannelNum, BSTR strParameter, long lPort, long windowNumber)

Interface IApplication

Last modified:

23-Oct-2008 Added support for N5264A
24-Apr-2008 Clarify ports with string names
23-Jul-2007 Added source port link
25-Apr-2007 Updated for ADC measurements.
12-Sept-2006 MQ Updated for logical receiver notation.

DataToMemory Method

Description	Stores the active measurement data into memory creating a memory trace. The memory can then be displayed or used in calculations with the measurement data.
VB Syntax	<code>meas.DataToMemory</code>
Variable	(Type) - Description
<code>meas</code>	A Measurement (object)
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>meas.DataToMemory</code>
C++ Syntax	HRESULT DataToMemory()
Interface	IMeasurement

Deembed Method

Description De-embeds a fixture from an existing Cal Set based on an S2P file. A new Cal Set is created with the effects of the fixture removed.

When the new Cal Set is applied to a channel, the effects of the fixturing are removed from the measurement data. Do NOT enable fixturing. The effects of the fixture are removed when the new Cal Set is selected and correction is turned ON.

VB Syntax `calMgr.Deembed (cs1,cs2,s2p,port, compPwr,extrap)`

Variable [\(Type\)](#) - Description

`calMgr` **(object)** - A [CalManager](#) object

`cs1` (String) Name of an existing Cal Set which resides on the PNA.

`cs2` (String) Name of new Cal Set which contains updated error terms with fixture de-embedded.

`s2p` (String) Name of the S2P file which characterizes the adapter/fixture.

`port` (Long Integer) Port number from which fixture will be de-embedded.

`compPwr` (Boolean)

True - When the Cal Set contains a power correction array for the fixture port, that array will be compensated for the fixture loss.

Warning: enabling power compensation can result in an increase in test port power and consequently, increased power to the DUT. Use with caution.

False - Do not compensate for loss in source power through the fixture.

`extrap` (Boolean)

True - Applies a simple extrapolation when the S2P file has a narrower frequency range than the Cal Set. The values for the first and last data points are extended in either direction to cover the frequency range of the Cal Set.

False - Extrapolation is NOT performed (default setting).

Return Type Not Applicable

Default Not Applicable

Example

```
calMgr.Deembed
"MyCalSet", "MyNewCalSet", "Fixture.s2p", 1, True, True
```

C++ Syntax HRESULT Deembed (BSTR srcSet, BSTR destSet,BSTR s2p, long port,
BOOL compPwr, BOOL extrap);

Interface ICalManager8

Last Modified:

25-Jan-2011 MX New topic

Delete Method

Description	Deletes the measurement.
VB Syntax	<i>meas.Delete</i>
Variable	(Type) - Description
<i>meas</i>	The Measurement object to delete (object)
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>meas.Delete</code>
C++ Syntax	HRESULT Delete()
Interface	IMeasurement

DeleteMarker Method

Description Deletes a marker from the measurement.

VB Syntax *meas.DeleteMarker(Mnum)*

Variable [\(Type\)](#) - Description

meas A Measurement (**object**)

Mnum (**long**) - Any existing marker number in the measurement

Return Type Not Applicable

Default Not Applicable

Examples `meas.DeleteMarker(1)`

C++ Syntax HRESULT DeleteMarker(long lMarkerNumber)

Interface IMeasurement

DeleteAllMarkers Method

Description	Deletes all of the markers from the measurement.
VB Syntax	<i>meas</i> .DeleteAllMarkers
Variable	(Type) - Description
<i>meas</i>	(object) - The Measurement object from which markers will be deleted.
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>meas.DeleteAllMarkers</code>
C++ Syntax	HRESULT DeleteAllMarkers()
Interface	IMeasurement

DeleteAllSegments Method

Description Removes all segments from the [scratch mixer](#).

VB Syntax *conv*.DeleteAllSegments()

Variable [\(Type\)](#) - Description

conv A [Converter Object](#)

Return Type Not Applicable

Default Not Applicable

Examples `mxr.DeleteAllSegments()`

[See example program](#)

C++ Syntax HRESULT DeleteAllSegments()

Interface IConverter5

Last Modified:

26-Oct-2010 New command (A.09.33)

DeleteCalSet Method

Description Deletes a Cal Set from the set of available Cal Sets. This method immediately updates the Cal Set file on the hard drive. If the Cal Set is currently being used by a channel or does not exist, this request will be denied and an error is returned.

Using the [Cal Sets collection](#) is a convenient way to manage Cal Sets.

VB Syntax *calMgr.DeleteCalSet (calset)*

Variable [\(Type\)](#) - Description

calMgr **(object)** - A [CalManager](#) object

calset **(string)** - Cal Set to be deleted. Specify the Cal Set by **GUID** or **Name**. Use [EnumerateCalSets](#) to list the available Cal Sets by name.

Return Type Not Applicable

Default Not Applicable

Example

```
Set pna=CreateObject("AgilentPNA835x.Application")
Set cmgr = pna.GetCalManager
cmgr.DeleteCalSet ("MyCalSet")
```

C++ Syntax HRESULT DeleteCalSet(BSTR strCalset);

Interface ICalManager

Last Modified:

6-Mar-2008 Added Name argument

DeleteConfiguration Method

Description	Deletes the specified configuration name from the PNA. The factory configurations cannot be deleted. This is the only method of programmatically distinguishing a factory configuration from a user-named configuration.
VB Syntax	<i>pathMgr.DeleteConfiguration name</i>
Variable	(Type) - Description
<i>pathMgr</i>	PathConfigurationManager (object)
<i>name</i>	(String) Configuration name to be deleted.
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>path.DeleteConfiguration "myMixer"</code>
C++ Syntax	HRESULT StoreConfiguration (long channelNum, BSTR configName);
Interface	IPathConfigurationManager

Last Modified:

14-Dec-2006 MX New topic

DeleteSegment Method

Description	Removes the specified number of segments from the scratch mixer starting at the index position.
VB Syntax	<i>conv</i> .DeleteSegment <i>index</i> , <i>count</i>
Variable	(Type) - Description
<i>conv</i>	A Converter Object
<i>index</i>	(Long integer) Position at which to start removing segments. Valid index range is between 1 and the current segment count. Use SegmentCount Property to read the current count in the Applied Mixer .
<i>count</i>	(Long integer) Optional argument. Number of segments to remove. If unspecified, 1 segment is removed.
Return Type	Not Applicable
Default	Not Applicable
Examples	<pre>mxr.DeleteSegment 1,5 'Removes 5 segments beginning at the first position.</pre> See example program
C++ Syntax	HRESULT DeleteSegment(long index, long count);
Interface	IConverter5

Last Modified:

26-Oct-2010 New command (A.09.33)

DeleteShortCut Method

Description Removes a macro from the list of macros in the analyzer. Does not remove the file.

Note: There are always 25 macro positions. They do not have to be sequential. For example, you can have number 7 but not numbers 1 to 6.

VB Syntax `app.DeleteShortCut item`

Variable [\(Type\)](#) - Description

app An [Application](#) (**object**)

item (**long integer**) number of the macro to be deleted.

Return Type Not Applicable

Default Not Applicable

Examples `app.DeleteShortCut 2`

C++ Syntax HRESULT DeleteShortcut(long Number)

Interface IApplication

Last Modified:

29-Apr-2013 Updated to 25 macros

DisallowAllEvents Method

Description Sets event filtering to monitor NO eventst.

VB Syntax *app*.**DisallowAllEvents**

Variable [\(Type\)](#) - Description

app An [Application](#) (**object**)

**Return
Type** Not Applicable

Default Not Applicable

Examples `app.DisallowAllEvents`

C++ Syntax HRESULT DisallowAllEvents()

Interface IApplication

Write only

DiscardChanges Method

Description	Cancels changes that have been made to the Converter setup and reverts to the previously-saved setup. Same as the Cancel button on the mixer setup dialog box .
VB Syntax	<i>conv</i> . DiscardChanges
Variable	(Type) - Description
<i>conv</i>	A Converter Object
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>conv.DiscardChanges</code>
C++ Syntax	HRESULT DiscardChanges();
Interface	IConverter

Last Modified:

2-Feb-2009 New topic

DisplayNAWindowDuringCalAcquisition Method

Description	<p>Set the 'show' state of the window to be displayed during a calibration.</p> <p>When this command is sent, the specified window is 'flagged' to be shown during calibration. The flag is cleared when the window is closed. A Preset or Instrument State Recall also closes the window. If the same window number is reopened, this command must be sent again to show the window during a calibration. The flag is NOT saved with an instrument state.</p> <p>Send this command for each additional window to show during a calibration.</p>
VB Syntax	<code>calMgr.DisplayNAWindowDuringCalAcquisition (winNum, State)</code>
Variable	(Type) - Description
<code>calMgr</code>	(object) - A CalManager object
<code>winNum</code>	(long) - Window number to show during a calibration. The calibration window will also be shown with this window.
	The window must already be created.
	Use NaWindows.count or app.WindowNumber to read existing window numbers.
<code>state</code>	(Boolean) Window state. Choose from:
	True - Show the specified window during calibration.
	False - Do NOT show the specified window during calibration.
Return Type	Not Applicable
Default	Not Applicable
Example	<pre>calMgr.DisplayNAWindowDuringCalAcquisition 2,True</pre>
	See example using this command
C++ Syntax	<code>HRESULT DisplayNAWindowDuringCalAcquisition(long WinNum, VARIANT_BOOL bVal);</code>
Interface	ICalManager5

28-Jan-2009 Removed 'Read'
8-Nov-2007 MX New topic

DisplayOnlyCalWindowDuringCalAcquisition Method

Description	Clears the flags for windows to be shown during calibrations other than the Cal Window. To flag a window to be shown see DisplayNAWindowDuringCalAcquisition
VB Syntax	<i>calMgr</i> . DisplayOnlyCalWindowDuringCalAcquisition
Variable	(Type) - Description
	<i>calMgr</i> (object) - A CalManager object
Return Type	Not Applicable
Default	Not Applicable
Example	<pre>calMgr.DisplayOnlyCalWindowDuringCalAcquisition</pre> See example using this command
C++ Syntax	HRESULT DisplayOnlyCalWindowDuringCalAcquisition()
Interface	ICalManager5

Last Modified:

8-Nov-2007 MX New topic

DoPrint Method

Description Prints the screen to the default Printer.

VB Syntax *app*.DoPrint

Variable [\(Type\)](#) - Description

app An [Application](#) (**object**)

Return Type Not Applicable

Default Not Applicable

Examples `app.DoPrint`

C++ Syntax HRESULT DoPrint()

Interface IApplication

DoECAL1PortEx Method

Description This method replaces [DoECAL1Port Method](#).

Does a 1-Port calibration using an ECAL module. You must first have a 1-port measurement active to perform the calibration.

The characterization within the ECal module that will be used for the calibration is specified by [ECALCharacterizationEx](#). The default value is 0.

VB Syntax *cal.DoECAL1PortEx [port][,module]*

Variable [\(Type\) - Description](#)

cal A Calibrator (**object**)

port (**long integer**) Optional argument - Port number to calibrate. Choose from:
1 - Calibrate port 1 (default if unspecified)
2 - Calibrate port 2

module (**long integer**) Optional argument. ECal module.

Choose from modules **1** through **8**

Use [IsECALModuleFoundEx](#) to determine the number of modules connected to the PNA

Use [GetECALModuleInfoEx](#) to returns the model and serial number of each module.

Return Type None

Default Not Applicable

Examples `cal.DoECAL1PortEx, 2, 2`

C++ Syntax HRESULT DoECAL1PortEx(long port, long moduleNumber = 1);

Interface ICalibrator4

DoECAL2PortEx Method

Description This method replaces [DoECAL2Port Method](#).

Does a 2-port calibration using an ECal module.

2-port refers to the number of ports to calibrate; NOT to the number of ECal module ports.

You must first have a measurement active to perform the calibration.

The characterization within the ECal module that will be used for the calibration is specified by [ECalCharacterizationEx](#). The default value is 0.

VB Syntax `cal.DoECAL2PortEx [portA][,portB][,module]`

Variable [\(Type\)](#) - Description

cal A Calibrator (**object**)

portA (**long integer**) Optional argument - Number of the receive port to calibrate.
Choose from:

- 1 - Calibrate port 1 (default, if unspecified)
- 2 - Calibrate port 2
- 3 - Calibrate port 3

And so forth for all available PNA / test set ports.

portB (**long integer**) Optional argument - Number of the source port to calibrate.
Choose from:

- 1 - Calibrate port 1
- 2 - Calibrate port 2 (default, if unspecified)
- 3 - Calibrate port 3

And so forth for all available PNA / test set ports.

module (**long integer**) Optional argument. ECal module.

Choose from modules **1** through **8**

Use [IsECALModuleFoundEx](#) to determine the number of modules connected to the PNA

Use [GetECALModuleInfoEx](#) to returns the model and serial number of each module.

Return Type None

Default Not Applicable

Examples `cal.DoECAL2PortEx,1,2,3`

C++ Syntax HRESULT DoECAL2PortEx(long portA = 1, long portB =2, long moduleNumber = 1);

Interface ICalibrator4

Last Modified:

1-Jan-2007 Corrected Port B default

DoneCalConfidenceCheckECAL Method

Description Concludes the Confidence Check and sets the ECal module back into the idle state.

VB Syntax *cal*.DoneCalConfidenceCheckECAL

Variable [\(Type\)](#) - Description

cal A Calibrator (**object**)

Return Type None

Default None

Examples *cal*.DoneCalConfidenceCheckECAL

C++ Syntax HRESULT DoneCalConfidenceCheckECAL();

Interface ICalibrator

DoReceiverPowerCal Method

Description	<p>Note: This command replaces DataToDivisor, LogMagnitudeOffset, Normalization, InterpolateNormalization.</p> <p>Immediately performs a receiver power calibration. The connection to the receiver must be in place when this command is sent.</p> <p>A Receiver Power Cal requires that the active measurement be an Unratioed power measurement.</p>
VB Syntax	<code>cal.DoReceiverPowerCal(param, srcPort [,pwrOffset])</code>
Variable	(Type) - Description
<i>cal</i>	A Calibrator (object)
<i>param</i>	(string) – Receiver to be calibrated. Choose any receiver in your PNA. See a block diagram of your PNA.
	<p>With PNA Rev 6.2, receivers can also be referred to using logical receiver notation. This notation makes it easy to refer to receivers with an external test set connected to the PNA. You do not need to know which physical receiver is used for each test port. Learn more.</p>
<i>srcPort</i>	(long integer) – Number of the port which will supply source power to the receiver during this cal.
	<p>Note: If the port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-x model, then you must use chan.getPortNumber to translate the string into a port number. To learn more see Remotely Specifying a Source Port.</p>
<i>pwrOffset</i>	(double) – Optional argument. Offset value in dB. Adjusts a receiver power cal to account for components or adapters that are added between the source port and receiver while performing this cal. Specify loss as a negative number; and gain as a positive number.
Return Type	None
Default	Not Applicable
Examples	<code>cal.DoReceiverPowerCal "B", 1, -10</code>
C++ Syntax	HRESULT DoReceiverPowerCal(BSTR parameter, long lSrcPort, double dPowerOffset);

Interface ICalibrator5

Last Modified:

24-Apr-2008	Added note for string names
30-Apr-2007	Edited for src strings

DoResponseCal Method

Description	Performs and immediately applies a Response cal. Same as selecting Normalize from the Unguided Cal - Measure Standards page. Learn more.
VB Syntax	<i>cal</i> .DoResponseCal (<i>measParam</i>),(<i>SourcePort</i>)
Variable	(Type) - Description
<i>cal</i>	A Calibrator (object)
<i>measParam</i>	(String) Measurement parameter to correct. It is NOT necessary for this measurement to be present.
<i>SourcePort</i>	(long integer) Source port number to calibrate. Optional for S-parameter measurements. Choose from: 0 - N5264A Measurement Receiver (no source ports). 1 - Calibrate port 1 2 - Calibrate port 2 (default, if unspecified) 3 - Calibrate port 3 And so forth for all available PNA / test set ports.
Return Type	None
Default	Not Applicable
Examples	<pre>cal.DoResponseCal "A/R",1</pre>
C++ Syntax	HRESULT DoResponseCal(BSTR param, long SourcePort);
Interface	ICalibrator9

Last Modified:

26-Oct-2009 MX New topic

Embed Method

Description Embeds a fixture (usually a matching network) into an existing Cal Set based on an S2P file. A new Cal Set is created with the effects of the matching network included in the correction data.

When the new Cal Set is applied to a channel, the effects of the fixture are included in the measurement data. Do NOT enable fixturing. The effects of the matching network are included when the new Cal Set is selected and correction is turned ON.

VB Syntax *calMgr*.**Embed** (*cs1,cs2,s2p,port, compPwr,extrap*)

Variable **(Type) - Description**

calMgr **(object)** - A [CalManager](#) object

cs1 (String) Name of an existing Cal Set which resides on the PNA.

cs2 (String) Name of new Cal Set which contains updated error terms with fixture embedded.

s2p (String) Name of the S2P file which characterizes the adapter/fixture.

port (Long Integer) Port number from which fixture will be embedded.

compPwr (Boolean)

True - Increase the source power to compensate for the loss through the fixture. The result is that the specified power level will be correct at the DUT input.

Warning: enabling power compensation can result in an increase in test port power and consequently, increased power to the DUT. Use with caution.

False - Do not compensate for loss in source power through the matching network.

extrap (Boolean)

True - Applies a simple extrapolation when the S2P file has a narrower frequency range than the Cal Set. The values for the first and last data points are extended in either direction to cover the frequency range of the Cal Set.

False - Extrapolation is NOT performed (default setting).

Return Type Not Applicable

Default Not Applicable

Example `calMgr.Embed "MyCalSet", "MyNewCalSet", "Fixture.s2p", 1, True, True`

C++ Syntax HRESULT Embed (BSTR srcSet, BSTR destSet, BSTR s2p, long port, BOOL compPwr, BOOL extrap);

Interface ICalManager8

Last Modified:

25-Jan-2011 MX New topic

EnumerateCalSets Method

Description Returns an array of Cal Set names being stored on the PNA.

VB Syntax *value* = *calMgr*.EnumerateCalSets

Variable [\(Type\)](#) - Description

value **(variant)** - Variable to store the returned Cal Set names

calMgr **(object)** - A [CalManager](#) object

Return Type VARIANT array

Default Not Applicable

Example

```
Dim pna
set
pna=CreateObject("AgilentPNA835x.Application")

Dim catalog
catalog=pna.getcalmanager.EnumerateCalSets
for i=lbound(catalog) to Ubound(catalog)
wscript.echo catalog(i)
next
```

C++ Syntax HRESULT EnumerateCalSets(VARIANT* names);

Interface ICalManager4

EnumerateItems Method

Description Returns a list of all name-value pairs (items) in the Cal Set.

See Also

[Item Property](#) (Learn about Name-Value pairs.)

[RemoveItem Method](#)

VB Syntax *names* = *CalSet*.**EnumerateItems**

Variable [\(Type\)](#) - Description

names (Variant array) List of string names.

CalSet **(object)** - A [CalSet](#) object

Return Type Variant

Default Not Applicable

Examples [See example](#)

C++ Syntax HRESULT EnumerateItems (VARIANT* itemNames);

Interface ICalSet6

Last Modified:

24-Sep-2010 MX New topic

Execute Method

Description Allows the use of COM to send a SCPI command.

This method can be used with :SYST:ERR? to convert scpi errors into text.

[See an example](#) of how to return error information when using the [Parse method](#).

Note: The SCPIStringParser Methods can NOT be used with [SCPI Status Reporting](#). However, the *OPC? will work.

VB Syntax Scpi.Execute (*SCPI_Command*)

Variable [\(Type\)](#) - Description

scpi A [ScpiStringParser](#) (**Object**)

SCPI_Command (**String**) - Any valid SCPI command

Return Type String

Default Not Applicable

Examples

```
Dim scpi As ScpiStringParser
Set scpi = app.ScpiStringParser
scpi.Execute("SYST:PRES");
ErrorString = scpi.Execute("SYST:ERRor?");
```

C++ Syntax Execute(BSTR SCPI_Command, BSTR * pQueryResponse);

Interface IScpiStringParser2

Last Modified:

27-Apr-2009 Added note

ExecuteShortcut Method

Description	Executes a Macro (shortcut) stored in the analyzer. Use app.getShortcut to list existing macros. Use app. putShortcut to associate the macro number with the file.
VB Syntax	<i>app.ExecuteShortcut index</i>
Variable	(Type) - Description
<i>app</i>	An Application (object)
<i>index</i>	(long integer) - Number of the macro stored in the analyzer.
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>app.ExecuteShortcut 1</code>
C++ Syntax	HRESULT ExecuteShortcut(long index)
Interface	IApplication

Read only

Exists Method

Description	Returns whether or not the specified Cal Set exists on the PNA.
VB Syntax	<i>calsets</i> . Exists (<i>string</i>)
Variable	(Type) - Description
<i>calsets</i>	A Calsets (collection)
<i>string</i>	(String) Name or GUID of the Cal Set enclosed in quotes.
Return Type	Boolean
	True - Cal Set exists
	False - Cal Set does NOT exist
Default	Not Applicable
Examples	<pre>dim check check=calsets.Exists "MyCalset" or check=calsets.Exists "7C4EEA5E-40D2-4D70-A048-33BFFE704163"</pre>
C++ Syntax	HRESULT Exists(BSTR nameOrGuid, VARIANT_BOOL * exists)
Interface	ICalSets2

Last Modified:

28-Feb-2011 New topic

Write-only

GenerateGlobalDeltaMatchSequence Method

Description	Initiates a global delta match calibration. Learn more about Delta match calibration. See example of a complete Delta Match calibration.
VB Syntax	<i>numSteps</i> = <i>guided</i> . GenerateGlobalDeltaMatchSequence <i>conn</i> , <i>cKit</i>
Variable	(Type) - Description
<i>numSteps</i>	Long Integer - Variable to store the returned number of connection steps required by the Global Delta Match Cal.
<i>guided</i>	GuidedCalibration (object)
<i>conn</i>	String Connector Type for port 1.
<i>cKit</i>	String Cal Kit for all ports.
Return Type	Not Applicable
Default	Not Applicable
Examples	<pre>guided.GenerateGlobalDeltaMatchSequence "APC 3.5 female", "85052B"</pre>
C++ Syntax	HRESULT GenerateGlobalDeltaMatchSequence(BSTR port_1_conn, BSTR cal_kit, long *num_steps);
Interface	IGuidedCalibration2

GenerateErrorTerms Method

Description Generates the error terms for the specified calibration type, stores the error terms in a Cal Set, saves the Cal Set, and returns the Cal Set GUID.

If ALL the data for the cal type has NOT been acquired an error message is returned.

Note: The manner in which the calibration is assigned to a Cal Set (Cal Register or User Cal Set) is determined by the setting of [RemoteCalStoragePreference](#).

VB Syntax *value* = *obj*.**GenerateErrorTerms**

Variable [\(Type\)](#) - Description

value (String) - Variable to store the returned GUID or error message.

obj Any of the following:

[GuidedCalibration](#) (object)

[SMCType](#) (object)

[VMCType](#) (object)

Return Type String

Default Not Applicable

Examples `string = SMC.GenerateErrorTerms`

C++ Syntax HRESULT GenerateErrorTerms(BSTR* calsetGUID);

Interface IGuidedCalibration

SMCType

VMCType

Last Modified:

28-Jan-2009 Fixed C++ Syntax

GenerateSteps Method

Description	Returns the number of steps required to complete the calibration. For an ECal User Characterization this generate steps for the ECal User Characterization process. The channel must already be calibrated using the same, or greater number of PNA ports as the ECal module. Also, the PNA ports must begin with Port 1 and use sequential port numbers. After this command is executed, subsequent commands can be used to query the number of measurement steps, issue the acquisition commands, query the connection description strings, and subsequently complete a User Characterization calibration.
VB Syntax	<i>value</i> = <i>obj</i> . GenerateSteps
Variable	(Type) - Description
<i>value</i>	(long) - Variable to store the returned number of steps
<i>obj</i>	Any of the following: GuidedCalibration (object) SMCType (object) VMCType (object) ECalUserCharacterizer (object)
Return Type	Long
Default	Not Applicable
Examples	<code>value = SMC.GenerateSteps</code>
C++ Syntax	HRESULT put_GenerateSteps(long* steps);
Interface	IGuidedCalibration SMCType VMCType IECalUserCharacterizer

4-Nov-2008 Added ECal object

GetAllSegments Method

Description Downloads a segment table from the PNA.

VB Syntax *segdata* = Segs.**GetAllSegments**

Variable [\(Type\)](#) - Description

segs A [Segments](#) (**Collection**)

segdata (Variant) A 2-dimensional array of Segment data:

- Dimension 0 is the number of elements in each segment.
- Dimension 1 is the number of segments that will be used.

All elements in the returned array are Variant. The type inside each Variant will be as is listed below.

The returned array will contain values for all elements regardless of the settings of [IFBandwidthOption](#), [SweepTimeOption](#), [SourcePowerOption](#) and [CouplePorts](#) properties. Ignore the values for the properties that are set to false.

The following is a list of dimension 0 elements for each segment:

- 0 = Segment state (Boolean True or False)
- 1 = Number of Points in this segment (Integer)
- 2 = Start Freq (Double)
- 3 = Stop Freq (Double)
- 4 = IFBW (Double)
- 5 = Dwell Time (Double)
- 6 + N = Power (Double) where N is the number of source ports of the PNA. For example, with a 4-port, 1-source PNA, indices 6 through 9 correspond to the per-segment power levels for Ports 1 to 4. Use [SourcePortCount Property](#) and [SourcePortNames Property](#) to see the available source ports for the PNA.

Return Type Variant, containing an array.

Default Not Applicable

Examples [See a VB example using this command](#)

[C++ Syntax](#) HRESULT GetAllSegments (VARIANT *pSegments);

[Interface](#) ISegments5

Last Modified:

28-Apr-2009 MX New topic

Read-only

GetAuxIO Method

Description This method returns the [IAuxIO](#) interface.

VB Syntax *app*.GetAuxIO

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

Return Type IHWAuxIO

Default Not Applicable

Example

```
Dim app As AgilentPNA835x.Application
Dim aux As HWAuxIO
Set aux = app.GetAuxIO
```

C++ Syntax HRESULT GetAuxIO (IHWAuxIO **pAux);

Interface IApplication

GetCalStandard Method

Description Returns a handle to a calibration standard for modifying its definitions. To select a standard for performing a calibration (use Calibrator.[AcquireCalStandard](#)).

VB Syntax *calkit*.GetCalStandard(*index*)

Variable [\(Type\)](#) - Description

calkit A calKit (**object**)

index (**long**) - Number of calibration standard. Choose **1** to **30**; (there are 30 cal standards in every kit).

Return Type calStandard

Default Not Applicable

Examples

```
Dim short As CalStandard
Set short = calKit.getCalStandard(1)
short.label = "myShort"
```

C++ Syntax HRESULT GetCalStandard(long standardNumber, ICalStandard **pCalStd)

Interface ICalKit

Read-only

GetCalKitTypeString Method

Description Returns ECal module model number and serial number based on the index number of the attached ECal modules.

VB Syntax *ECalID* = *cal*.**GetCalKitTypeString** (*module*)

Variable [\(Type\)](#) - Description

ECalID **(string)** - variable to store the returned ECal module ID information.

cal A [Calibrator](#) **(object)**

module **(long integer)** ECal module.

Choose from modules **1** through **8**

Use [IsECALModuleFoundEx](#) to determine the number of modules connected to the PNA

Return Type String

Default Not Applicable

Examples `info = cal.GetCalKitTypeString(2)`

Example return string:

```
"N4691-60003 ECal 01234"
```

C++ Syntax HRESULT GetCalKitTypeString(long moduleNumber, BSTR* info);

Interface ICalibrator8

Last Modified:

30-Oct-2009 MX New topic

GetCompatibleCalKits Method

Description **Note:** This command replaces [CompatibleCalKits Property](#)

Returns a comma-separated list of valid kits that use the specified connector type. This includes mechanical cal kits, applicable characterizations found within ECal modules currently connected to the PNA, and all user characterizations stored in PNA disk memory.

For ECal modules, the returned list includes the serial numbers.

See the [ECalUserCharacterizer Object](#).

Use items in the list to select the kit to be used with the [CalKitType Property](#)

VB Syntax *value* = *guidCal*.**GetCompatibleCalKits** (*connectorType*)

Variable **(Type) - Description**

value (Variant) - Variable to store the returned list of cal kits. One-dimensional array of string values.

guidCal A [GuidedCalibration](#) (object)

connectorType (String) Connector type for which compatible cal kits will be returned.

Use [ValidConnectorType](#) to return a list of connector type strings.

Use [ConnectorType](#) to set the connector type for each port to be calibrated.

Return Type Variant – Containing one-dimensional array of strings.

Default Not Applicable

Examples

```
Dim kits As Variant
kits = guidedCal.GetCompatibleCalKits "Type N (50) male"
```

C++ Syntax HRESULT GetCompatibleCalKits(BSTR connector, VARIANT* Kits);

Interface IGuidedCalibration5

Last Modified:

17-Feb-2011 Edited connector type argument

25-Aug-2009 MX New topic

GetCalManager Method

Description This method returns the [ICalManager](#) interface.

VB Syntax *app*.GetCalManager()

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

Return Type ICalManager*

Default Not Applicable

Example

```
dim app as AgilentPNA835x.Application
dim mgr as CalManager
set mgr = app.GetCalManager()
```

C++ Syntax HRESULT GetCalManager(ICalManager
**mgr);

Interface IApplication

Last Modified:

30-Jan-2008 Added parenthesis

Get CalSetByGUID Method

Description Requests a Cal Set by GUID. Returns an ICal Set interface.

VB Syntax *calMgr*.**GetCalSetByGUID** (*GUID*)

Variable [\(Type\)](#) - Description

calMgr **(object)** - A CalManager object

GUID **(string)** - GUID of the Cal Set being requested.

Return Type Interface object

Default Not Applicable

Example `calMgr.GetCalSetByGUID (2B893E7A-971A-11d5-8D6C-00108334AE96)`

C++ Syntax HRESULT GetCalSetByGUID(BSTR* strGUID, ICal Set* pCalSet);

Interface ICalManager

GetCalSetCatalog Method - Superseded

Description	This method is replaced with EnumerateCalSets Returns a string containing a list of comma-separated GUIDs in the following format: {FD6F863E-9719-11d5-8D6C-00108334AE96}, {1B03B2CE-971A-11d5-8D6C-00108334AE96}, {2B893E7A-971A-11d5-8D6C-00108334AE96}
VB Syntax	<i>value</i> = <i>calMgr</i> . GetCalSetCatalog
Variable	(Type) - Description
<i>value</i>	(string) - Variable to store the returned GUID list
<i>calMgr</i>	(object) - A CalManager object
Return Type	String
Default	Not Applicable
Example	<pre>value = calMgr.GetCalSetCatalog</pre>
C++ Syntax	HRESULT GetCalSetCatalog(BSTR);
Interface	ICalManager

Last Modified:

6-Mar-2008 Superseded

GetCalSetUsageInfo Method

Description Returns a string identifying the Cal Set currently in use by the specified channel.

This method identifies the Cal Set being used by returning its GUID.

This method also identifies the "Error Term set" within the Cal Set.

Error term sets are identified by integers, with set 0 belonging to the original (non-interpolated) terms. As stimulus values for a channel are changed causing interpolation to be required, a new Error Term set is constructed within the Cal Set to hold the interpolated Error Terms. The sets are sequentially numbered 1, 2, 3, and so forth. These Error Term sets are destroyed when they are no longer being used.

If there is no Cal Set in use for the given channel, the <GUID> argument is set to the empty string.

VB Syntax *calMgr*.**GetCalSetUsageInfo** *chan*, *GUID*, *setNumber*

Variable [\(Type\) - Description](#)

calMgr **(object)** - A CalManager object

chan **(long)** - channel of the Cal Set being requested

GUID **(string)** - variable to store the GUID of the Cal Set being requested. If there is no Cal Set in use for the given channel, the <GUID> argument is set to the empty string.

setNumber **(long)** - variable to store the error term ID being requested. If the returned argument is greater than 0, the set is being interpolated.

Return Type String, Long Integer

Default Not Applicable

Example `calMgr.GetCalSetUsageInfo 1, GUID, EtermID`

C++ Syntax HRESULT GetCalSetUsageInfo (long lChannel, BSTR* CalSetGUID, long* etermSetID);

Interface ICalManager

13-May-2011 Removed parens

GetCalTypes Method

Description Returns a list of available calibration types known to the PNA. The Standard CalTypes are the same on all PNA's, but the Custom CalTypes are not necessarily the same. They are dependent on the custom measurement in the PNA. [Learn more about applying Cal Types.](#)

See also [CalibrationTypeID](#) to apply a Cal Type containing in a Cal Set.

VB Syntax `v = mgr.GetCalTypes`

Variable [\(Type\)](#) - Description

mgr A [CalManager](#) (**Object**)

v Name/GuidPair that contains the calibration type name and associated GUID for each cal type known to the PNA.

Return Type (**variant**) Two dimensional array.

Default Not Applicable

Examples `v = CalManager.GetCalTypes`

C++ Syntax HRESULT GetCalTypes(VARIANT * NameGuidPair)

Interface ICalManager2

GetComplex Method

Description	Retrieves complex data from the specified location. See also getNAComplex , getData , and getPairedData Methods
VB Syntax	<i>measData</i> . getComplex <i>location</i> , <i>numPts</i> , <i>real</i> () , <i>imag</i> ()
Variable	(Type) - Description
<i>measData</i>	An IArrayTransfer interface which supports the Measurement object
<i>location</i>	(enum NADataStore - IArrayTransfer) - Where the data you want is residing. Choose from: 0 - naRawData 1 - naCorrectedData 2 - naMeasResult 3 - naRawMemory 4 - naMemoryResult 5 - naDivisor - When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using DataToDivisor Method .
<i>numPts</i>	(long integer) - Number of data points requested [out] - specifies number of data elements returned [in] - specifies the data being requested or the capacity of the arrays
<i>real</i>	(single) - Array to store the real values
<i>imag</i>	(single) - Array to store the imaginary values
Return Type	Single
Default	Not Applicable
Examples	<pre>Dim real(201) AS Single Dim imag(201) AS Single Dim pts as Integer Dim measData As IArrayTransfer Set measData = app.ActiveMeasurement measData.getComplex naCorrectedData, pts, real(0), imag(0)</pre>
C++ Syntax	IArrayTransfer - HRESULT getComplex(tagNADataStore DataStore, long* pNumValues, float* pReal, float* pImag)

Interface IArrayTransfer

GetConnectedPhaseReferences Method

Description	Reads the ID strings of the phase references that are currently connected to the PNA USB.
VB Syntax	<i>refs</i> = <i>phaseRef</i> .GetConnectedPhaseReferences
Variable	(Type) - Description
<i>phaseRef</i>	A PhaseReferenceCalibration (object)
Return Type	Not Applicable
Default	Not Applicable
Examples	<pre>ref= phaseRef.GetConnectedPhaseReferences()</pre> See example program
C++ Syntax	HRESULT GetConnectedPhaseReferences()
Interface	IPhaseReferenceCalibration

Last modified:

3-Apr-2012 New topic

GetConverter Method

Description This method returns a handle to a [Converter](#) object.

VB Syntax *chan*.GetConverter()

Variable [\(Type\)](#) - Description

chan A [Channel](#) (object)

Return Type IConverter

Default Not Applicable

Example

```
Dim app as AgilentPNA835x.Application

Set app =
CreateObject("AgilentPNA835x.Application")

Dim chan As IChannel

Set chan = app.ActiveChannel

Dim convert

Set convert = chan.GetConverter()
```

C++ Syntax HRESULT GetConverter(IConverter **obj);

Interface IChannel

Last Modified:

2-Feb-2009 New topic

getDataByString Method

Description Retrieves variant data from the specified location in your choice of formats.

The PNA returns complex trace data which is ratioed if required by the measurement parameter, such as S11 or A/B. Otherwise it is raw receiver data, such as A or B.

Equation Editor Notes:

- When equation editor is active on a trace in a standard S-parameter channel, GetData returns the data from the parameter on the trace that was measured last. For example, for the equation "S22 + S33 + S11", then S33 is the last measured parameter because it uses source port 3.
- In [applications](#), if equation editor is active and the original parameter for the trace is not requested anywhere in the channel, then zeros are returned. If the original parameter is being measured within the channel, then data for the original parameter is returned.
- In general, if an equation contains no measurement parameters, then data for the original parameter is returned.

VB Syntax *data = meas.getDataByString location, format*

Variable [\(Type\)](#) - Description

data **(variant)** - Array to store the data.

meas **(object)** - A Measurement object

location **(string)** – Name of the buffer to be read. Choose from:

"naRawData"

"naCorrectedData"

"naMeasResult"

"naRawMemory"

"naMemoryResult"

"naDivisor" - When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using [DataToDivisor Method](#).

See [Data Access Map](#)

format (**enum NADataFormat**) - Format in which you would like the data. It does not have to be the displayed format. Choose from:

- 0 - naDataFormat_LinMag
- 1 - naDataFormat_LogMag
- 2 - naDataFormat_Phase
- 3 - naDataFormat_Polar
- 4 - naDataFormat_Smith
- 5 - naDataFormat_Delay
- 6 - naDataFormat_Real
- 7 - naDataFormat_Imaginary
- 8 - naDataFormat_SWR
- 9 - naDataFormat_PhaseUnwrapped
- 10 - naDataFormat_InverseSmith
- 11 - naDataFormat_Kelvin
- 12 - naDataFormat_Fahrenheit
- 13 - naDataFormat_Centigrade

[Learn more about Data Format.](#)

* Specify **Smith** or **Polar** formats to obtain complex data pairs, which require a two-dimensional array **varData (numpts, 2)** to accommodate both real and imaginary data.

All scalar formats return a single dimension **varData(numpts)**.

Return Type Variant array

Default Not Applicable

Examples `meas.getDataByString "naMeasResult", naDataFormat_Phase`

C++ Syntax HRESULT getDataByString(BSTR location, tagDataFormat dataFormat, VARIANT * pData);

Interface IMeasurement

Last Modified:

11-Jun-
2009

Added EE notes

1-Oct-2007

Added temperature formats

19-Jul-2007

Corrected example

GetData Method

Description Retrieves variant data from the specified location in your choice of formats. To get smoothed data from any of the specified locations, the format must be the same as the displayed format.

The PNA returns complex trace data which is ratioed if required by the measurement parameter, such as S11 or A/B. Otherwise it is raw receiver data, such as A or B.

This method returns a variant which is less efficient than methods available on the [IArrayTransfer interface](#).

If you plan to **Put** this data back into analyzer, [putDataComplex](#) (variant data) method requires complex, two-dimensional data. Therefore, request the data in **Polar** format.

Equation Editor Notes:

- When equation editor is active on a trace in a standard S-parameter channel, GetData returns the data from the parameter on the trace that was measured last. For example, for the equation "S22 + S33 + S11", then S33 is the last measured parameter because it uses source port 3.
- In [applications](#), if equation editor is active and the original parameter for the trace is not requested anywhere in the channel, then zeros are returned. If the original parameter is being measured within the channel, then data for the original parameter is returned.
- In general, if an equation contains no measurement parameters, then data for the original parameter is returned.

VB Syntax *data = meas.GetData location, format*

Variable [\(Type\) - Description](#)

data Variant array to store the data.

meas A Measurement (**object**)

location (**enum NADataStore**) - Where the data you want is residing. See [Data Access Map](#). Choose from:

0 - naRawData

1 - naCorrectedData

2 - naMeasResult

3 - naRawMemory

4 - naMemoryResult

5 - naDivisor When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using [DataToDivisor Method](#).

format (**enum NADataFormat**) - Format in which you would like the data. It does not have to be the displayed format. Choose from:

0 - naDataFormat_LinMag

1 - naDataFormat_LogMag

2 - naDataFormat_Phase

3 - naDataFormat_Polar*

4 - naDataFormat_Smith*

5 - naDataFormat_Delay

6 - naDataFormat_Real

7 - naDataFormat_Imaginary

8 - naDataFormat_SWR

9 - naDataFormat_PhaseUnwrapped

10 - naDataFormat_InverseSmith

11 - naDataFormat_Kelvin

12 - naDataFormat_Fahrenheit

13 - naDataFormat_Centigrade

[Learn more about Data Format.](#)

* **Specify Smith or Polar** formats to obtain complex data pairs, which require a two-dimensional array **varData (numpts, 2)** to accommodate both real and imaginary data.

All scalar formats return a single dimension **varData(numpts)**.

naDataFormat_Phase and **naDataFormat_PhaseUnwrapped** returns degrees. However, [putDataScalar](#) method accepts data in radians (not degrees) and displays in degrees.

Return Type Variant array - automatically dimensioned to the size of the data

Default Not Applicable

Examples

```
Dim varData As Variant
varData = meas.GetData(naMeasResult,naDataFormat_Phase)
'Print Data
For i = 0 to chan.NumberOfPoints-1
  Print varData(i)
Next i
```

[See a C# example.](#)

C++ Syntax HRESULT getData(tagNADataStore DataStore, tagDataFormat DataFormat, VARIANT *pData)

Interface IMeasurement

Last Modified:

11-Jun-2009	Added EE notes
14-Apr-2009	Added link to C#
1-Oct-2007	Added temperature formats

GetECALModuleInfoEx Method

Description	<p>This property replaces Get ECALModuleInfo Method.</p> <p>Returns the following information about the connected ECAL module: model number, serial number, connector type, calibration date, min and max frequency.</p> <p>The characterization within the ECal module that this information will be read from is specified by ECALCharacterizationEx. The default value is 0.</p>
VB Syntax	<i>moduleInfo</i> = <i>cal</i> . GetECALModuleInfoEx (<i>module</i>)
Variable	(Type) - Description
<i>moduleInfo</i>	(string) - variable to store the module information
<i>cal</i>	A Calibrator (object)
<i>module</i>	(long integer) ECal module.
	Choose from modules 1 through 8
	Use IsECALModuleFoundEx to determine the number of modules connected to the PNA
Return Type	String
Default	Not Applicable
Examples	<pre>info = cal.GetECALModuleInfoEx(2)</pre>
	Example return string:
	<pre>ModelNumber: 85092-60007, SerialNumber: 01386, ConnectorType: N5FN5F, PortAConnector: Type N (50) female, PortBConnector: Type N (50) female, MinFreq: 30000, MaxFreq: 9100000000, NumberOfPoints: 250, Calibrated: July 4 2002</pre>
C++ Syntax	<code>HRESULT GetECALModuleInfoEx(long moduleNumber, BSTR* info);</code>
Interface	ICalibrator4

GetEcalUserCharacterizer Method

Description This method returns a handle to an [ECalUserCharacterizer](#) object.

VB Syntax *calMgr*.GetEcalUserCharacterizer()

Variable [\(Type\)](#) - Description

calMgr A [CalManager Object](#) (**object**)

Return Type IEcalUserCharacterizer

Default Not Applicable

Example

```
Dim mgr as ICalManager
Set mgr = app.GetCalManager
Dim ecalCharacterizer
Set ecalCharacterizer =
mgr.GetEcalUserCharacterizer()
```

C++ Syntax HRESULT GetEcalUserCharacterizer(IEcalUserCharacterizer
**obj);

Interface ICalManager6

Last Modified:

6-Mar-2009 MX New topic

GetENRData Method

Description	Read the ENR calibration data from PNA memory.
VB Syntax	<i>vData</i> = <i>enr</i> . GetENRData ()
Variable	(Type) - Description
<i>vData</i>	Variable to store the returned ENR data. Frequency value in Hz, followed by corresponding ENR value in dB.
<i>enr</i>	An ENRFile (object)
Return Type	Variant Array
Default	Not Applicable
Examples	See example program
C++ Syntax	HRESULT GetENRData (VARIANT vdata);
Interface	IENRFile

Last Modified:

2-Aug-2007 MX New topic

GetErrorCorrection Method

Description Reads the error correction state for the channel.

Use [ErrorCorrection Property](#) to set this value.

When this command returns true, some measurements on the channel MAY not have error correction ON. This is because the Cal Set currently in place may not contain the appropriate calibration data. To read the error correction state for a measurement, use [Error Correction Property](#).

VB Syntax *chan*.GetErrorCorrection (*boolean*)

Variable [\(Type\)](#) - Description

chan A [Channel](#) (**object**)

boolean (**boolean**) Variable to store the returned value.

False - Error correction has been set OFF

True - Error correction has been set ON

Return Type Boolean

Default [About Error Correction](#)

Examples `chan.GetErrorCorrection(value)`

C++ Syntax HRESULT GetErrorCorrection (VARIANT_BOOL *bState)

Interface IChannel8

GetErrorTerm Method - Superseded

Description **Note:** This command is replaced by [Get ErrorTermByString_Method](#)

Retrieves error term data that is used for error correction. The data is complex pairs. Memory for the returned Variant is allocated by the server. The server returns a variant containing a two-dimensional safe Array.

This method returns a variant which is less efficient than [getErrorTermComplex](#) on the ICalData interface.

Note: When performing an ECal, send [SetCalInfoEx Method](#) BEFORE calling GetErrorTerm method.

[Learn about reading and writing Calibration data.](#)

VB Syntax `data = cal.getErrorTerm term, rcv, src`

Variable [\(Type\)](#) - Description

<i>data</i>	Variant array to store the data.
<i>cal</i>	A Calibrator (object)
<i>term</i>	(enum As NaErrorTerm). Choose from: naErrorTerm_Directivity_Isolation naErrorTerm_Match naErrorTerm_Tracking
<i>rcv</i>	(long integer) - Receiver Port
<i>src</i>	(long integer) - Source Port

To get this	Specify these parameters:		
Error Term	<i>term</i>	<i>rcv</i>	<i>src</i>
Fwd Directivity	naET_Directivity Isolation	1	1
Rev Directivity	naET_Directivity Isolation	2	2
Fwd Isolation	naET_Directivity Isolation	2	1
Rev Isolation	naET_Directivity Isolation	1	2
Fwd Source Match	naErrorTerm_Match	1	1
Rev Source Match	naErrorTerm_Match	2	2
Fwd Load Match	naErrorTerm_Match	2	1
Rev Load Match	naErrorTerm_Match	1	2
Fwd Reflection Tracking	naErrorTerm_Tracking	1	1
Rev Reflection Tracking	naErrorTerm_Tracking	2	2
Fwd Trans Tracking	naErrorTerm_Tracking	2	1
Rev Trans Tracking	naErrorTerm_Tracking	1	2

Return Type Variant

Default Not Applicable

Examples

```
Dim varError As Variant
varError = cal.getErrorTerm(naErrorTerm_Tracking,2,1)
```

C++ Syntax HRESULT getErrorTerm(tagNAErrorTerm ETerm, long ReceivePort, long SourcePort, VARIANT* pData)

Interface ICalibrator

GetErrorTerm Method **Superseded**

Description	<p>This command has been replaced with GetErrorTermByString</p> <p>Returns error term data from the Cal Set. The returned data is complex pairs.</p> <p>Learn more about Reading and Writing Cal Data</p> <p>See examples of Reading and Writing Cal Set Data</p>
VB Syntax	<i>data</i> = <i>calSet</i> . getErrorTerm (<i>setNumber</i> , <i>term</i> , <i>rcv</i> , <i>src</i>)
Variable	(Type) - Description
<i>data</i>	(Variant) Two-dimensional safe array to store the returned data. Memory for the returned Variant is allocated by the PNA and must be released by client.
	Note: See also getErrorTermComplex on the ICalData2 interface to avoid using the variant data type.
<i>calSet</i>	A Cal Set (object)
<i>setNumber</i>	(Long) There can be more than one set of error terms in a Cal Set. <ul style="list-style-type: none"> • SetNumber 0 contains the original set of error terms for a Cal Set. • SetNumbers > 0 contain Interpolated error terms. Interpolated error terms are generated when interpolation is required and destroyed when no longer used. Learn about Interpolation. • To determine the SetNumber in use by a channel, see GetCalSetUsageInfo
<i>term</i>	(enum As NaErrorTerm2). Choose from: <ul style="list-style-type: none"> 0 - naET_Directivity (rcv = src) 1 - naET_SourceMatch (rcv = src) 2 - naET_ReflectionTracking (rcv = src) 3 - naET_TransmissionTracking (rcv ≠ src) 4 - naET_LoadMatch (rcv ≠ src) 5 - naET_Isolation (rcv ≠ src)
<i>rcv</i>	(Long) - Receiver Port
<i>src</i>	(Long) - Source Port

Return Type Variant

Default Not Applicable

Examples

```
Dim varError As Variant  
varError = CalSet.getErrorTerm(0,naET_TransmissionTracking,2,1)
```

C++ Syntax HRESULT getErrorTerm(long setID, tagNAErrorTerm2 ETerm, long ReceivePort, long SourcePort, VARIANT* pData)

Interface ICalSet

GetErrorTermByString Method

Description Returns error term data from the Cal Set by specifying the string name of the error term.

- Learn more about [Reading and Writing Cal Data](#)
- See examples of [Reading](#) and [Writing](#) Cal Set Data
- See [GetCalSetUsageInfo](#) to determine the setNumber.

VB Syntax `pdata = calset.GetErrorTermByString(setNumber, errorTerm)`

Variable [\(Type\)](#) - Description

pdata **(Variant)** Two-dimensional safe array to store the returned data. Memory for the returned Variant is allocated by the PNA and must be released by client.

Note: See also [getErrorTermComplexByString](#) on the ICalData3 interface to avoid using the variant data type.

calset A [Cal Set](#) **(object)**

setNumber **(Long)** Set number of the required Cal Set data.

See [GetCalSetUsageInfo](#) to determine the setNumber.

- SetNumber **0** contains the original "master" set of error terms for a Cal Set.
- SetNumbers **> 0** refers to the PNA channel number that contains the error terms. When retrieving channel error terms, Correction must be ON.

New beginning PNA Rev 7.2: The channel error term data contains [interpolation](#), [fixturing](#), and [port extension](#) data if each is ON.

- For Balanced Measurements, interpolation, fixturing, and port extensions can be ON independently.
- For Standard S-parameters, to get port extension data, both fixturing and port extensions must be ON.

errorTerm **(String)** The string name used to identify a particular error term in the Cal Set. An example string for port 3 directivity in a full 2 port cal might be "Directivity(3,3)". To determine the string names of error terms, see [GetErrorTermList2](#).

Return Type Variant

Default Not Applicable

Examples [See an Example](#)

C++ Syntax HRESULT GetErrorTermByString (long SetNumber, BSTR bufferName, VARIANT* pdata);

Interface ICalSet2

GetErrorTermComplex Method **Superseded**

Description	<p>This command has been replaced by GetErrorTermComplexByString</p> <p>Retrieves error term data from the error correction buffer. The data is in complex pairs.</p> <p>Note: When performing an ECal, send SetCalInfoEx Method BEFORE calling GetErrorTermComplex method.</p> <p>Learn more about reading and writing Cal Data using COM.</p> <p>This method exists on a non-default interface. If you cannot access this method, use the GetErrorTerm Method on ICalibrator.</p>
VB Syntax	<i>eData</i> .GetErrorTermComplex <i>term</i> , <i>rcv</i> , <i>src</i> , <i>numPts</i> , <i>real</i> (<i></i>), <i>imag</i> (<i></i>)
Variable	(Type) - Description
<i>eData</i>	An ICalData pointer to the Calibrator object
<i>term</i>	(enum NAErrorTerm) - The error term to be retrieved. Choose from: <ul style="list-style-type: none"> • naErrorTerm_Directivity_Isolation • naErrorTerm_Match • naErrorTerm_Tracking
<i>rcv</i>	(long integer) - Receiver Port
<i>src</i>	(long integer) - Source Port
<i>numPts</i>	(long integer) - on input, max number of data points to return; on output: indicates the actual number of data points returned.
<i>real</i> (<i></i>)	(single) - array to accept the real part of the error-term. One-dimensional for the number of data points.
<i>imag</i> (<i></i>)	(single) - array to accept the imaginary part of the error-term. One-dimensional for the number of data points.

	To get this	Specify these parameters:		
	Error Term	<i>term</i>	<i>rcv</i>	<i>src</i>
	Fwd Directivity	naET_Directivity Isolation	1	1
	Rev Directivity	naET_Directivity Isolation	2	2
	Fwd Isolation	naET_Directivity Isolation	2	1
	Rev Isolation	naET_Directivity Isolation	1	2
	Fwd Source Match	naErrorTerm_Match	1	1
	Rev Source Match	naErrorTerm_Match	2	2
	Fwd Load Match	naErrorTerm_Match	2	1
	Rev Load Match	naErrorTerm_Match	1	2
	Fwd Reflection Tracking	naErrorTerm_Tracking	1	1
	Rev Reflection Tracking	naErrorTerm_Tracking	2	2
	Fwd Trans Tracking	naErrorTerm_Tracking	2	1
	Rev Trans Tracking	naErrorTerm_Tracking	1	2
Return Type	Single			
Default	Not Applicable			
Examples	<pre> ReDim rel(numpts) ReDim img(numpts) Dim eData As ICalData Set eData = chan.Calibrator eData.getErrorTermComplex naErrorTerm_Directivity_Isolation, 1, 1, 201, rel(0), img(0) </pre>			
C++ Syntax	HRESULT raw_getErrorTermComplex(tagNAErrorTerm ETerm, long ReceivePort, long SourcePort, long* pNumValues, float* pReal, float* pImag)			
Interface	ICalData			

GetErrorTermComplex Method **Superseded**

Description This command is [replaced](#) with [Get ErrorTermComplexByString](#)

Returns error term data from the Cal Set. The data is in complex pairs.

Learn more about [Reading and Writing Cal Data](#)

See examples of [Reading](#) and [Writing](#) Cal Set Data

Note: This method exists on a non-default interface. If you cannot access this method, use the [GetErrorTerm](#) Method on ICal Set.

VB Syntax `iCalData2.GetErrorTermComplex setNumber, term, rcv, src, numPts, real(), imag()`

Variable [\(Type\) - Description](#)

iCalData2 An [ICalData2](#) pointer to the Cal Set object

setNumber **(Long)** There can be more than one set of error terms in a Cal Set.

- setNumber **0** contains the original set of error terms for a Cal Set.
- setNumbers **> 0** contain Interpolated error terms. Interpolated error terms are generated when interpolation is required and destroyed when no longer used. [Learn about Interpolation.](#)
- To determine the setNumber in use by a channel, see [GetCalSetUsageInfo](#)

term **(enum NAErrorTerm2)** - The error term to be retrieved. Choose from:

- 0 - naET_Directivity
- 1 - naET_SourceMatch
- 2 - naET_ReflectionTracking
- 3 - naET_TransmissionTracking
- 4 - naET_LoadMatch
- 5 - naET_Isolation

rcv **(Long)** - Receiver Port

src **(Long)** - Source Port

numPts **(Long)** An In/Out parameter.

On the way **in**, you specify the **max** number of values being requested.

On the way **out**, the PNA returns number of values actually returned.

real() **(single)** - array to accept the **real** part of the error-term. One-dimensional for the number of data points.

imag() **(single)** - array to accept the **imaginary** part of the error-term. One-dimensional for the number of data points.

Return Type Single

Default Not Applicable

Examples

```
dim numpts as long
numpts = ActiveChannel.NumberOfPoints
ReDim r(numpts) ' real part
ReDim i(numpts) ' imaginary part
Dim CalSet as CalSet
set CalSet = pna.GetCalManager.GetCal SetByGUID( txtGUID )
Dim eData As ICalData2
Set eData = CalSet
eData.getErrorTermComplex 0, naET_LoadMatch, 1, 2, numpts,
r(0),i (0)
```

C++ Syntax HRESULT getErrorTermComplex(long setID, tagNAErrorTerm2 ETerm, long ReceivePort, long SourcePort, long* pNumValues, float* pReal, float* pImag)

Interface ICalData2

GetErrorTermComplexByString Method

Description Returns error term data from the Cal Set by specifying the string name.

Learn more about [Reading and Writing Cal Data](#)

See examples of [Reading](#) and [Writing](#) Cal Set Data

Note: This method exists on a non-default interface. If you cannot access this method, use [GetErrorTermByString](#)

VB Syntax *ICalData3*.**GetErrorTermComplexByString** *setNumber*, *errorTerm*, *numPoints*, *real(0)*, *imag(0)*

Variable [\(Type\)](#) - [Description](#)

ICalData3 An ICalData3 pointer to a [CalSet](#) (Object)

setNumber **(Long)** There can be more than one set of error terms in a Cal Set.

- *setNumber* **0** contains the original set of error terms for a Cal Set.
- *setNumbers* **> 0** contain Interpolated error terms. Interpolated error terms are generated when interpolation is required and destroyed when no longer used. [Learn about Interpolation.](#)
- To determine the *setNumber* in use by a channel, see [GetCalSetUsageInfo](#)

errorTerm **(String)** The string name of error term in the Cal Set. An example string for port 3 directivity in a full 2 port cal might be "Directivity(3,3)".

For a list error term string names, use [Get ErrorTermList2](#)

numPoints **(Long)** An In/Out parameter.

On the way **in**, you specify the **max** number of values being requested.

On the way **out**, the PNA returns number of values actually returned.

real **(Single)** The real component of the complex data.

imag **(Single)** The imaginary component of the complex data.

Return Type Single

Default Not Applicable

Examples [See example](#)

C++ Syntax HRESULT GetErrorTermComplexByString(long etermSetID, BSTR
bufferName, long* lnumPoints, single* real, single* imag);

Interface ICalData3

GetErrorTermList Method **Superseded**

Description	<p>Note: This command is replaced by CalSet.getErrorTermList2</p> <p>Returns the list of Error Terms contained in this Cal Set for the CalType specified in the OpenCal Set method. Learn more about reading and writing Cal Data using COM.</p> <p>The list is a comma separated, textual representation of the error terms with the term name followed by the port path in parentheses: Term (n, n), Term (m,n)</p> <p>Before calling this method you must open the Cal Set with OpenCal Set. If the Cal set is not open, this method returns E_NA_Cal Set_ACCESS_DENIED.</p> <p>Use StringToNAErrorTerm2 to convert the list entrees to values that can be used with GetErrorTerm and PutErrorTerm.</p> <p>Note: The port path designation (m n) indicates the ports that contribute to the error being compensated. Directivity, source match and reflection tracking are single port characteristics, designated in this list by (n n) where n equals the port being characterized. Other terms characterize the interaction between ports. For example, the load match term is describing the match at port (m) while looking into port (n). Thus the notation (m n) indicates the two ports that contribute to the loadmatch error.</p>
VB Syntax	<i>CalSet</i> . GetErrorTermList (SetID, count, strList)
Variable	(Type) - Description
<i>CalSet</i>	(object) - A Cal Set object
<i>SetID</i>	(long) - specifies the error term set to query. Use 0 for the master set.
<i>count</i>	(long) - the number of error terms in the returned list
<i>strList</i>	(string) - comma separated list of error terms found in Cal Set
Return Type	Not Applicable
Default	Not Applicable
Examples	<pre>dim count as Integer dim list as string OpenCalSet (naCalType_TwoPortSOLT 1, 2) GetErrorTermList(0, count, list)</pre>

```
CloseCalSet( )
```

Assuming the cal set contained the full set of error terms for this two-port Cal, the returned list would be:

```
"Directivity(1 1),SourceMatch(1 1),ReflectionTracking(1  
1),TransmissionTracking(2 1),LoadMatch(2 1),Isolation(2  
1),Directivity(2 2),SourceMatch(2 2),ReflectionTracking(2  
2),TransmissionTracking(1 2),LoadMatch(1 2),Isolation(1 2)"
```

[C++ Syntax](#) HRESULT GetErrorTermList (long etermSetID, long* count, BSTR* strList);

[Interface](#) ICalSet

GetErrorTermList2 Method

Description Returns a list of error terms names found in the Cal Set containing the specified prefix.

Learn more about [Reading and Writing Cal Data](#)

See examples of [Reading](#) and [Writing](#) Cal Set Data

VB Syntax *list* = *CalSet*.**GetErrorTermList2**(*setNumber*, *calTypePrefix*)

Variable [\(Type\) - Description](#)

list **(Variant)** Variant containing a string array of error term names.

CalSet **(object)** - A [CalSet](#) object

setNumber **(Long)** There can be more than one set of error terms in a Cal Set.

- *setNumber* **0** contains the original set of error terms for a Cal Set.
- *setNumbers* **> 0** contain Interpolated error terms. Interpolated error terms are generated when interpolation is required and destroyed when no longer used. [Learn about Interpolation.](#)
- To determine the *setNumber* in use by a channel, see [GetCalSetUsageInfo](#)

caltypePrefix **(String)** The string used to identify Cal Set data as belonging to a specific Cal Type. This string is used as a filter so that only the error term names of interest are returned. If the prefix is empty, all terms are returned.

An example prefix for a two port cal on ports 2 and 3 might be: "Full 2 Port Cal (2,3)".

Return Type Variant

Default Not Applicable

Examples [See an Example](#)

C++ Syntax HRESULT GetErrorTermList2 (long SetNumber, BSTR caltypePrefix, VARIANT* list)

Interface ICalSet2

GetErrorTermStimulus Method

Description Returns the stimulus values over which the specific error term was acquired. For example, with mixer channels, you may get a different set of values for Directivity at the input port versus Directivity at the output port.

- Learn more about [Reading and Writing Cal Data](#)
- See examples of [Reading](#) and [Writing](#) Cal Set Data
- See [GetCalSetUsageInfo](#) to determine the setNumber.
- See [PutErrorTermStimulus Method](#).

VB Syntax `pdata = calset.GetErrorTermStimulus(setNumber, bufferName)`

Variable [\(Type\)](#) - Description

pdata **(Variant)** The PNA will allocate the memory for the returned variant. The data is returned as a SafeArray of Variant elements. Each element is of VarType double.

calset A [Cal Set](#) **(object)**

setNumber **(Long)** Set number of the required Cal Set data.

See [GetCalSetUsageInfo](#) to determine the setNumber.

- SetNumber **0** contains the original "master" set of error terms for a Cal Set.
- SetNumbers **> 0** refers to the PNA channel number that contains the error terms. When retrieving channel error terms, Correction must be ON. The channel error term data **may** be [interpolated](#) and **may** also be [compensating for a fixture](#) if that feature is on.
 - For Balanced Measurements, interpolation, fixturing, and port extensions can be ON independently.
 - For Standard S-parameters, to get port extension data, both fixturing and port extensions must be ON.

bufferName **(String)** The string name used to identify a particular error term in the Cal Set. An example string for port 3 directivity in a full 2 port cal might be "Directivity(3,3)". To determine the string names of error terms, see [GetErrorTermList2](#).

Return Type Variant

Default Not Applicable

Examples The sequence is:

```
complexData = calset1.GetErrorTermByString(0, BufferName)
```

```
frequencyData = calset1.GetErrorTermStimulus(0, BufferName)
```

```
// manipulate complex data here
```

```
Calset2.PutErrorTermByString(BufferName, manipulatedComplexData)
```

```
Calset2.PutErrorTermStimulus(BufferName, frequencyData);
```

[See an Example](#)

C++ Syntax HRESULT GetErrorTermStimulus (long SetNumber, BSTR bufferName, VARIANT* pdata);

Interface ICalSet7

Last modified:

2-May-2011 New topic

Write-only

GetExtendedCalInterface Method

Description Returns an interface that exposes the properties of Noise Calibration.

VB Syntax *Cal2*.**GetExtendedCalInterface** (*interface*)

Variable [\(Type\) - Description](#)

Cal2 An ICalibrate2 (**object**)

interface (object) Returns a handle to the specified interface. Choose from '

"NoiseCal"

Return Type

Default

Example

```
dim noiseCal
dim noiseCalExtensions
set noiseCal= Get Calmanager?.CreateCustomCalEx("NoiseCal")
set noiseCalExtensions =
noiseCal.GetExtendedCalInterface("INoiseCal")
```

C++ Syntax HRESULT GetExtendedCalInterface();

Interface ICalibrate2

Last Modified:

29-May-2007 MN New topic

Get ExternalTestSetIO Method

Description This method returns the [IExternalTestSetIO](#) interface.

VB Syntax *app*.GetExternalTestSetIO

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

Return Type IHWExternalTestSetIO

Default Not Applicable

Example

```
Dim app As AgilentPNA835x.Application
Dim ets As HWExternalTestSetIO
Set ets = app.GetExternalTestSetIO
```

C++ Syntax HRESULT GetExternalTestSetIO (IHWExternalTestSetIO
**ptestset);

Interface IApplication

GetFilterStatistics Method

Description Returns all four Filter Statistics resulting from a [SearchFilterBandwidth](#). To retrieve individual filter statistics, use [meas.FilterCF](#), [meas.FilterBW](#), [meas.FilterLoss](#), [meas.FilterQ](#) properties.

VB Syntax `meas.GetFilterStatistics cf,bw,loss,q`

Variable [\(Type\)](#) - Description

`meas` A Measurement (**object**)

`cf,bw,loss,q` Dimensioned variables to store the returned values

Return Type **(double)** `cf`
(single) `bw,loss,q`

Default Not Applicable

Examples

```
'Dimension variables
Dim cf as Double
Dim bw as Single
Dim loss as Single
Dim q as Single

meas.GetFilterStatistics cf,bw,loss,q
```

C++ Syntax HRESULT GetFilterStatistics(double* centerFreq, float* bw, float* loss, float* quality)

Interface IMeasurement

GetGuid Method

Description Returns a string containing the GUID identifying this Cal Set. Each Cal Set is assigned a GUID (global unique ID). GUIDs are used to retrieve and select Cal Sets on the PNA. Learn more about [reading and writing Cal Data using COM](#).

VB Syntax *value* = *CalSet*.**GetGuid**

Variable [\(Type\)](#) - Description

value **(string)** - Variable to store the returned GUID

CalSet **(object)** - A Cal Set object

Return Type String

Default Not Applicable

Examples `guid = CalSet.GetGuid 'Read`

C++ Syntax HRESULT GetGUID(BSTR* pGUIDString);

Interface ICalSet

InputVoltageEX Property

Description	This command replaces get InputVoltage Method Reads the ADC voltage from the specified location.
VB Syntax	<code>volts = AuxIO.InputVoltageEX loc</code>
Variable	(Type) - Description
<i>volts</i>	(double) - variable to store the return value
<i>AuxIO</i>	(object) - A Hardware Auxiliary Input / Output object
<i>loc</i>	(Long) Location from which to read data. For PNA-X models: , reads ADC voltages from the Power I/O connector . Choose from: <ol style="list-style-type: none"> 1 Reads voltage on Analog In 1 port (pin 7). 2 Reads voltage on Analog In 2 port (pin 8). 3 Reads voltage on GndSens (pin 6). 4 Reads voltage on Analog Out 1 port (pin 3). 5 Reads voltage on Analog Out 2 port (pin 4). For all other PNA models: <ol style="list-style-type: none"> 1 Reads voltage on the Analog IN (pin 14) of the AUX IO connector. 4 Reads voltage on Analog Out 1 port (pin 3). 5 Reads voltage on Analog Out 2 port (pin 2).
Return Type	Double
Default	Not Applicable
Examples	<pre>Dim aux as HWAuxIO Set aux = PNA.getAuxIO volts = aux.InputVoltageEX 1 'for PNA-X, read voltage on PowerI/O pin 7 'for all other models, reads voltage on Aux I/O Analog In (pin 14)</pre>
C++ Syntax	HRESULT get_InputVoltageEX (long muxLoc, double* vtVoltage);

Interface HWAuxIO2

Last Modified:

5-Aug-2008	Removed 'get'
10-Jul-2007	MX New topic

get_Input1 Method

Description	<p>Reads a hardware latch that captures high to low transitions on Input1 of the Material Handler IO. Reading the latch causes it to reset and is ready for the next transition. The hardware latch is only capable of capturing one transition per query. Additional transitions are ignored until after the next query.</p> <p>Momentarily driving Input1 high, then low, causes a transition to be detected and latched.</p>
VB Syntax	<pre><i>inp1</i> = handlerIo.get_Input1</pre>
Variable	(Type) - Description
<i>inp1</i>	(variant) - A variable to store the return value
<i>handlerIo</i>	(object) - A HandlerIO object
Return Type	Variant -
	0 - a high to low transition occurred at Input1 since the last time it was queried.
	1 - no high to low transition occurred.
Default	Not Applicable
Examples	<pre>input1 = handlerIo.get_Input1 'Read</pre>
C++ Syntax	HRESULT get_Input1 (VARIANT* Data);
Interface	IHWMaterialHandlerIO

GetIPConfigurationStruct Method

Description	Returns an NA_IPConfiguration data structure which contains information about the current status of the PNA's computer networking configuration. This is the same set of information that is returned in a single string by the LANConfiguration property.
VB Syntax	<i>value</i> = <i>app</i> .GetIPConfigurationStruct
Variable	(Type) - Description
<i>value</i>	(NA_IPConfiguration) Variable to receive the PNA IP (LAN) configuration information.
<i>app</i>	An Application (object)
Return Type	NA_IPConfiguration
Default	Not Applicable

Examples

```
Dim networkConfigInfo As NA_IPConfiguration
networkConfigInfo = app.GetIPConfigurationStruct()

MsgBox "Host name = " & networkConfigInfo.HostName

MsgBox "Domain name = " & networkConfigInfo.DomainName

MsgBox "IP address = " & networkConfigInfo.IPAddress

If Not networkConfigInfo.DHCPEnabled Then
    MsgBox "IP address is static"
Else
    MsgBox "IP address is dynamic"
End If

MsgBox "Subnet mask = " & networkConfigInfo.SubNet

MsgBox "Gateway = " & networkConfigInfo.DefaultGateway

MsgBox "Primary DNS server = " & networkConfigInfo.DNSServer1

MsgBox "Secondary DNS server = " & networkConfigInfo.DNSServer2

MsgBox "First suffix in DNS suffix search order = " &
networkConfigInfo.DNSSuffix1

MsgBox "Second suffix in DNS suffix search order = " &
networkConfigInfo.DNSSuffix2

MsgBox "Primary WINS server = " &
```

```
networkConfigInfo.PrimaryWINSServer  
  
MsgBox "Secondary WINS server = " &  
networkConfigInfo.SecondaryWINSServer  
  
MsgBox "Network adapter device ID = " &  
networkConfigInfo.DeviceID  
  
MsgBox "Description of network adapter = " &  
networkConfigInfo.Description  
  
MsgBox "MAC address = " & networkConfigInfo.MacAddress
```

C++ Syntax HRESULT GetIPConfigurationStruct (tagNA_IPConfiguration * pIPConfig);

Interface IApplication14

Last Modified:

2-Jun-2008 MX New topic

GetIsolationPaths Method

Description Gets the list of paths (port pairings) for which isolation standards will be measured during calibration.

VB Syntax *value* = *obj*.**GetIsolationPaths**

Variable [\(Type\)](#) - **Description**

value (Variant) - Variable to store the returned port paths in pairs. One-dimensional array of Long Integers.

obj Any of the following:

[GuidedCalibration](#) (object)

Return Type Variant – Containing one-dimensional array of Long Integers.

Default No port pairs (empty Variant variable)

Examples

```
pathList = guidedCal.GetIsolationPaths
'displaying the paths separated by commas,
'with a dash (-) between the pair of port numbers comprising
each path

For i = LBound(portList) To UBound(portList) Step 2
    msg = msg + CStr(portList(i)) + "-" + CStr(portList(i+1))
    If i+1 < UBound(portList) Then msg = msg + ","
Next
MsgBox msg, 0, "List of isolation paths"
```

C++ Syntax HRESULT GetIsolationPaths(VARIANT* pathList);

Interface IGuidedCalibration

Last Modified:

16-Apr-2007 MX New topic

GetLibraryFunctions Method

Description	Returns the functions in an imported (loaded) DLL.
VB Syntax	<i>functions</i> = <i>equation</i> . GetLibraryFunctions <i>location</i>
Variable	(Type) - Description
<i>functions</i>	(variant) - Array to store the returned functions.
<i>equation</i>	A MeasurementEquation object
<i>location</i>	(string) – Full path and filename of the *.dll to be read.
Return Type	Variant array
Default	Not Applicable
Examples	<pre>functions=equation.GetLibraryFunctions "C:/Program Files/Agilent/Network Analyzer/UserFunctions/Expansion.dll"</pre>
C++ Syntax	HRESULT GetLibraryFunctions(BSTR filename, BSTR* functionList);
Interface	IMeasurementEquation2

Last Modified:

10-Jan-2011 New topic

Get MaterialHandlerIO Method

Description This method returns the [MaterialHandlerIO](#) interface.

VB Syntax *app*.**GetMaterialHandlerIO**

Variable [\(Type\)](#) - Description

app An [Application](#) (**object**)

Return Type IHWMaterialHandlerIO

Default Not Applicable

Example

```
Dim app As AgilentPNA835x.Application
Dim hand As HWMaterialHandlerIO
Set hand = app.GetMaterialHandlerIO
```

C++ Syntax HRESULT GetMaterialHandlerIO (IHWMaterialHandlerIO
**phand);

Interface IApplication

GetNAComplex Method

Description	Retrieves complex data from the specified location. See also getComplex and getData Method.
VB Syntax	<i>measData</i> .getNAComplex <i>location</i> , <i>numPts</i> , <i>data</i>
Variable	(Type) - Description
<i>measData</i>	An IArrayTransfer interface which supports the Measurement object
<i>location</i>	(enum NADataStore) - Where the data you want is residing. Choose from: 0 - naRawData 1 - naCorrectedData 2 - naMeasResult 3 - naRawMemory 4 - naMemoryResult 5 - naDivisor - When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using DataToDivisor Method .
<i>numPts</i>	(long integer) - Number of data points requested [out] - specifies number of data elements returned [in] - specifies the data being requested or the capacity of the <i>dComplex</i> array
<i>data</i>	(NAComplex) - A one-dimensional array of NaComplex to store the data.
Return Type	NAComplex
Default	Not Applicable
Examples	<pre>Dim dComplex(201) AS NaComplex Dim measData As IArrayTransfer Dim pts as Long Set measData = app.ActiveMeasurement measData.getNAComplex naCorrectedData, pts, dComplex(0)</pre>
Notes	The data is stored as Real and Imaginary (Re and Im) members of the NaComplex user defined type. You can access each number individually by iterating through the array. <pre>For i = 0 to NumPts-1 dReal (i) = dcomplex (i).Re dImag (i) = dcomplex (i).Im</pre>

Next i

C++ Syntax HRESULT getNAComplex(tagNADataStore DataStore, long* pNumValues, TsComplex* pComplex)

Interface IArrayTransfer

GetNumberOfGroups Method

Description	Returns the number of groups a channel has yet to acquire. To set the number of groups for a channel, use Number Of Groups Method
VB Syntax	<i>value</i> = <i>chan</i> .GetNumberOfGroups
Variable	(Type) - Description
<i>value</i>	(Long Integer) - Number of groups
<i>chan</i>	Channel (object)
Return Type	(Long Integer)
Default	Not Applicable
Examples	<code>groups = chan.GetNumberOfGroups 'Read</code>
C++ Syntax	HRESULT GetNumberOfGroups(long* numberOfGroups);
Interface	IChannel3

get_Output Method

Description [Type 1 and Type2 configurations](#): Returns the last value written to the selected output pin.

[Type3 configuration](#): Returns the current state of the selected output pin. If an Input1 trigger occurs, the state may not be the same value as was written.

All configurations: Data is written using [put_Output](#) Method.

VB Syntax `data = handlerIo.get_Output (pin)`

Variable [\(Type\) - Description](#)

data **(variant)** - A variable to store the return value. The returned value will be one of the following:

0 - TTL Low

1 - TTL High

handlerIo **(object)** - A HandlerIO object

pin **(enum as NAMatHandlerOutput)** - output to read. Choose from:

naOutput1 (0)

naOutput1User (1)

naOutput2 (2)

naOutput2User (3)

[Learn about User Output](#)

Return Type Variant

Default Not Applicable

Examples `data = handlerIo.get_Output (naOutput1)`

C++ Syntax `HRESULT get_Output (tagNAMatHandlerOutput Output, VARIANT* Data);`

Interface IHWMaterialHandlerIO

Read-only

get_OutputVoltage Method

Description **E836x and PNA-L:** Reads voltages on the DAC/Analog Output 1|2 of the Auxiliary IO connector.

PNA-X: Reads voltage on the [Power I/O connector](#) AnalogOut1|2.

VB Syntax *volts* = *AuxIO*.get_OutputVoltage (*output*)

Variable [\(Type\)](#) - Description

volts **(double)** - variable to store the return value

AuxIO **(object)** - A Hardware Auxiliary Input / Output object

output **(variant)** Number of the output DAC from which to read voltage. Choose from:

1 Output 1 (Aux I/O pin 3) and (Power I/O pin 3)

2 Output 2 (Aux I/O pin 2) and (Power I/O pin 4)

Return Type Double

Default Not Applicable

Examples

```
Dim aux as HWAuxIO
Set aux = PNA.getAuxIO
volts = aux.get_OutputVoltage(1)
'read voltage from Analog Out 1 (Aux I/O pin3) or (Power I/O pin 3)
```

C++ Syntax HRESULT get_OutputVoltage(VARIANT Output, double* Voltage);

Interface IHWAuxIO

Last Modified:

10-Jul-2007 Added PNA-X capability

get OutputVoltageMode Method

Description	This command returns the mode of the selected "Analog Out" line on the Power I/O connector . The modes give the user the option to have the requested voltage applied immediately or not until the sweep is done. To set the mode, use put_OutputVoltageMode Method .
VB Syntax	<i>mode</i> = <i>auxIo</i> . get_OutputVoltageMode (<i>output</i>)
Variable	(Type) - Description
<i>mode</i>	(enum NAOutputVoltageMode) - variable to store the returned mode.
	naWaitEOS - While in this mode any voltage changes sent to the selected analog out will only get applied to the output between sweeps.
	naNoWait - While in this mode any voltage changes sent to the selected analog out will occur right away without waiting until the end of a sweep, the voltage gets applied immediately.
<i>auxIo</i>	(object) - A Hardware Auxiliary Input / Output object
<i>output</i>	(double) Analog Output. Choose from 1 or 2
Return Type	enum as NAOutputVoltageMode
Default	naWaitEOS
Examples	<pre>vOutMode = auxIo.get_OutputVoltageMode (1)</pre>
C++ Syntax	HRESULT get_OutputVoltageMode(VARIANT Output, tagNAOutputVoltageMode* pMode);
Interface	IHWAuxIO

Last Modified:

10-Jul-2007 Added PNA-X capability

GetPairedData Method

Description Retrieves pairs of data from the specified location.

Note: This method exists on a non-default interface. If you cannot access this method, use the [Get Data Method](#) on IMeasurement.

VB Syntax `measData.getPairedData location, format, numPts, d1, d2`

Variable (Type) - Description

measData An IArrayTransfer interface which supports the Measurement object

location (**enum NADataStore**) - Where the data you want is residing. Choose from:

0 - naRawData

1 - naCorrectedData

2 - naMeasResult

3 - naRawMemory

4 - naMemoryResult

5 - naDivisor - When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using [DataToDivisor Method](#).

See [Data Access Map](#)

format (**enum NAPairedDataFormat**) - Format in which you would like the Paired data. Choose from:

0 -naLogMagPhase - Log magnitude and phase

1 -naLinMagPhase - Linear magnitude and phase

2 -naRealImaginary - Real and Imaginary

Note: Selecting **naRealImaginary** format is the same as using the [getComplex](#) method

numPts (**long integer**) - Number of data points requested

[out] - specifies number of data elements returned

[in] - specifies the data being requested or the capacity of the *dPaired* array

d1 (**single**) - Array to store the magnitude / real values

d2 (single) - Array to store the phase / imaginary values

Return Type Two Single arrays

Default Not Applicable

Examples

```
Dim logm() As Single
Dim phase() As Single
Public measData As IArrayTransfer
Set measData = app.ActiveMeasurement
Dim numpts As Long
numPoints = app.ActiveChannel.NumberOfPoints
ReDim logm(numPoints)
ReDim phase(numPoints)

measData.getPairedData naCorrectedData, naLogMagPhase,
numPoints, logm(0), phase(0)

Print values(0), values(1)
```

C++ Syntax HRESULT getPairedData(tagNADataStore DataStore,
tagNAPairedDataFormat PairFormat, long* pNumValues, float* pReal, float*
pImag)

Interface IArrayTransfer

get_Port Method

Description Returns the value from the specified "readable" port.

VB Syntax `data = handlerIo.get_Port (port)`

Variable [\(Type\) - Description](#)

data (**variant**) - A variable to store the return value. The following table shows what the returned data represents:

Port	MSB.....LSB 8.....0
C	C3...C0
D	D3...D0
E	D3...D0 + C3...C0

handlerIo (**object**) - A HandlerIO object

port (**enum as NAMatHandlerPort**) - port to get data from. Choose from:

naPortC - (2)

naPortD - (3)

naPortE - (4)

Note: Reading data from the Write-only ports (A,B,F,G,H) will return an error. Ports C and D must be put in Read mode before reading from C, D, or E using [PortMode Property](#).

Return Type Variant

Default 0

Examples `data = handlerIo.get_Port(naPortC)`

C++ Syntax `HRESULT get_Port (tagNAMatHandlerPort Port, VARIANT* Data);`

Interface IHWMaterialHandlerIO

Read-only

get_PortCData Method

Description Reads a 4-bit value from Port C of the Aux I/O connector (pins 22-25) and the Material Handler IO (pins 21-24 Anritsu) - (pins 22-25 Avantest).

Note: These lines are connected to both the Handler IO and Aux IO in the PNA.

VB Syntax *value* = *AuxIO*.get_PortCData

Variable [\(Type\)](#) - Description

value **(variant)** - Variable to store the returned data

AuxIO **(object)** - A Hardware Auxiliary Input / Output object

Return Type Integer

Default None

Examples `value = auxIo.get_PortCData 'Reading a value of 15 when in Positive Logic indicates Port C lines C0, C1, C2, C3 are High. If in Negative Logic they are Low.'`

C++ Syntax HRESULT get_PortCData(VARIANT* Data);

Interface IHWAuxIO

GetPortNumber Method

Description Returns the port number that is associated with the specified port name. These numbers are used with several commands to specify a PNA port.

To learn more, see [Remotely Specifying a Source Port](#).

VB Syntax *value* = *object*.**GetPortNumber** (*portName*)

Variable (Type) - Description

value **(Long)** - Variable to store the returned Port Number integer value.

object A [Channel](#) **(object)** - always more complete than capabilities object.

A [Capabilities](#) **(object)** - use when a channel is not available

portName **(String)** Name of the PNA port.

- Use [SourcePortNames Property](#) to return a list of PNA port (string) names.
- If an external source is selected, specify the external source name that is used in the [Select an External Source dialog](#).

Return Type Long Integer

Default Not Applicable

Examples `value = chan.GetPortNumber ("Src2 Out1") 'Read`

C++ Syntax HRESULT GetPortNumber(BSTR name, long *number);

Interface IChannel13

ICapabilities4

Last Modified:

12-Oct-2009	Added parens
23-May-2008	Added channel object
24-Apr-2008	Fixed example
30-Apr-2007	Edited for Ext Source Control

GetRaw2DData Method

Description Returns raw data at all frequency and power data points for **any** GCA sweep. Previously 2D sweep only.

- When using SMART sweep, ALL data is returned including ALL background iteration sweeps. Use [TotalIterations](#) to determine the number of iteration sweeps. The number of data points that are returned is always the number of frequency points times the number of iteration sweeps.
- When using 2D sweeps, ALL data is returned. The number of data points returned / freq may vary. [Learn more.](#)

Use the standard "[get data](#)" commands to return just the displayed data results (not the background sweeps).

A compression parameter must be present. [Learn more.](#)

VB Syntax `data = gca.GetRaw2DData location, format, param`

Variable [\(Type\)](#) - Description

data Variant array in which to store returned measurement data.

gca A [GainCompression](#) (**object**)

location (enum NADataStore) - Where the data you want is residing. Choose from:

0 - naRawData

1 - naCorrectedData

format (enum NADataFormat) - Format in which you would like the data. It does not have to be the displayed format. Choose from:

0 - naDataFormat_LinMag

1 - naDataFormat_LogMag

2 - naDataFormat_Phase

3 - naDataFormat_Polar*

4 - naDataFormat_Smith*

5 - naDataFormat_Delay -- **Not valid for this command.**

6 - naDataFormat_Real

7 - naDataFormat_Imaginary

8 - naDataFormat_SWR

9 - naDataFormat_PhaseUnwrapped

10 - naDataFormat_InverseSmith

[Learn more about Data Format.](#)

* Specify Smith or Polar formats to obtain complex data pairs, which require a two-dimensional array varData (numpts, 2) to accommodate both real and imaginary data.

All scalar formats return a single dimension varData(numpts).

naDataFormat_Phase and naDataFormat_PhaseUnwrapped returns degrees.

param **(String)** Parameter of data to return. Not case-sensitive. The specified parameter need NOT be displayed. However, a compression parameter must be present. [Learn more.](#)

Choose from:

- **"pin"** - (CompIn21) Input power at the compression point.
- **"pout"** - (CompOut21) Output power at the compression point.
- **"gain"** - (CompGain21) Device gain (S21) at the compression point.
- **"inputmatch"** - (CompS11) Input match at the compression point.
- **"DeltaGain"** - (DeltaGain21) Measured Gain (watts) / Ref Gain (watts). [Learn more.](#)
- **"AI1"** and **"AI2"** - ADC measurements at the specified compression level. [Learn more.](#)

Return Type Variant Array

Default Not Applicable

Examples `data = gca.GetRaw2DData naRawData, naDataFormat_Real, "pin"`

C++ Syntax HRESULT GetRaw2DData(tagNADataStore location, tagNADataFormat format, BSTR data_name, VARIANT* pData);

Interface IGainCompression

Last Modified:

9-May-2012	Edited Smart text
30-Aug-2011	Added 2D sweep variation
13-May-2011	Added returns background sweeps
31-Mar-2009	Added new params
12-May-2008	Edited for any GCA sweep
22-Oct-2007	MX New topic

GetDataIm Method

- Description** For a specified data point, returns the imaginary part of the specified Gain Compression data. If correction is on, corrected data are returned. Otherwise, raw data are returned. Can be used with Smart and 2D sweeps.
- For SMART sweep, the number of data points that are returned is always going to be the number of iteration sweeps. Use `IterationSweeps` to determine the number of iteration sweeps.
 - For 2D sweeps, the number of data points returned / freq may vary. [Learn more.](#)

VB Syntax `data = gca.GetDataIm stim, dPoint, param`

Variable (Type) - Description

`data` Variant array in which to store returned measurement data.

`gca` A [GainCompression](#) (object)

`stim` (**NAGCAIndexSelect**)

- **naFrequencySelect** - for the specified frequency data point, returns all of the measured data for each power stimulus.
- **naPowerSelect** - for the specified power data point, returns all of the measured data for each frequency stimulus.

`dPoint` Data point (Frequency or Power) for which data is returned.

`param` Parameter of data to return. Not case-sensitive. Choose from:

- **"pin"** - input power at each data point.
- **"pout"** - output power at each data point.
- **"gain"** - device gain (S21) at each data point.
- **"inputmatch"** - input match (S11) at each data point.
- **"DeltaGain"** - Measured Gain (watts) / Ref Gain (watts). [Learn more.](#)
- **"AI1"** and **"AI2"** - ADC measurements at the specified compression level. [Learn more.](#)

Return Type Variant Array

Default Not Applicable

Examples For the fifth frequency data point, returns 'Power Output' imaginary (phase) data from all power stimulus values. If there are 30 power sweep points, 30 values are returned.

```
data = gca.GetDataIm naFrequencySelect, 5, "pout"
```

For the 30th stimulus power data point, returns 'Power Output' imaginary (phase) data from all frequency stimulus values. If there are 201 frequency sweep points, 201 values are returned.

```
data = gca.GetDataIm naPowerSelect, 30, "pout"
```

Note: For 2D sweeps, the number of data points returned / freq may vary.

[Learn more.](#)

C++ Syntax HRESULT GetDataIm(tagNAGCAIndexSelect index_select, int index, BSTR data_name, VARIANT* pData);

Interface IGainCompression

Last Modified:

9-May-2012	Edited for Smart sweep
30-Aug-2011	Added 2D sweep variation
31-Mar-2009	Added new params
12-May-2008	Edited for any GCA sweep
22-Oct-2007	MX New topic

GetDataRe Method

Description Reads the REAL part of the data acquired from **any** GCA sweep. Previously 2D sweep only.

Note: For 2D sweeps, the number of data points returned / freq may vary.
[Learn more.](#)

VB Syntax `data = gca.GetDataRe stim, dPoint, param`

Variable [\(Type\)](#) - Description

data Variant array in which to store returned measurement data.

gca A [GainCompression](#) (object)

stim (NAGCAIndexSelect)

- **naFrequencySelect** - for the specified frequency data point, returns all of the measured data for each power stimulus.
- **naPowerSelect** - for the specified power data point, returns all of the measured data for each frequency stimulus.

dPoint Data point (Frequency or Power) for which data is returned.

param Parameter of data to return. Not case-sensitive. Choose from:

- **"pin"** - input power at each data point.
- **"pout"** - output power at each data point.
- **"gain"** - device gain (S21) at each data point.
- **"inputmatch"** - input match (S11) at each data point.
- **"DeltaGain"** - Measured Gain (watts) / Ref Gain (watts). [Learn more.](#)
- **"AI1"** and **"AI2"** - ADC measurements at the specified compression level. [Learn more.](#)

Return Type Variant Array

Default Not Applicable

Examples For the fifth frequency data point, returns 'Power Output' REAL data from all power stimulus values. If there are 30 power sweep points, 30 values are returned.

```
data = gca.GetDataRe naFrequencySelect, 5, "pout"
```

For the 30th stimulus power data point, returns 'Power Output' REAL data from all frequency stimulus values. If there are 201 frequency sweep points, 201 values are returned.

```
data = gca.GetDataRe naPowerSelect, 30, "pout"
```

Note: For 2D sweeps, the number of data points returned / freq may vary.
[Learn more.](#)

C++ Syntax HRESULT GetDataRe(tagNAGCAIndexSelect index_select, int index, BSTR data_name, VARIANT* pData);

Interface IGainCompression

Last Modified:

30-Aug-2011	Added 2D sweep variation
31-Mar-2009	Added new params
12-May-2008	Edited for any GCA sweep
22-Oct-2007	MX New topic

GetRxLevelingConfiguration Method

Description This method returns a handle to a [RxLevelingConfiguration](#) object.

VB Syntax *chan*.**GetRxLevelingConfiguration()**

Variable [\(Type\)](#) - Description

chan A [Channel](#) (object)

Return Type IRxLevelingConfiguration

Default Not Applicable

Example

```
dim app as AgilentPNA835x.Application
dim mgr as RxLevelingConfiguration
set mgr = app.GetRxLevelingConfiguration()
```

C++ Syntax HRESULT GetRxLevelingConfiguration(IRxLevelingConfiguration
**mgr);

Interface IChannel17

Last Modified:

3-Mar-2009 MX New topic

GetSourceByRole Method

Description Returns the name of a source that is assigned to the specified role.

Note: For non-converter channels, use [chan.RoleDevice](#)

VB Syntax *source* = *conv*.**GetSourceByRole** (*role*)

Variable [\(Type\)](#) - Description

source (String) Source name, from [Source Configuration dialog](#), that is assigned to the specified role.

conv A [Converter Object](#)

role (String) Role for which the source name will be returned. Use [GetSourceRoles](#) for a list of valid roles.

Return Type Not Applicable

Default Not Applicable

Examples `RF2Source = conv.GetSourceByRole ("RF2")`

C++ Syntax HRESULT GetSourceByRole(BSTR roleID, BSTR deviceName);

Interface IConverter

Last Modified:

3-Jan-2012 Remove superseded (thx SW)

3-May-2011 superseded

2-Feb-2009 New topic

Read only

GetSourceRoles Method

Description Returns the defined role names ("RF2", "LO1").

Note: For non-converter channels, use [chan.RoleDevice](#)

VB Syntax *roles* = *conv*.**GetSourceRoles** ()

Variable [\(Type\)](#) - Description

roles (Variant array) Variable to store returned list of valid roles.

conv A [Converter Object](#)

Return Type Variant

Default Not Applicable

Examples `roles = conv.GetSourceRoles()`

C++ Syntax HRESULT GetSourceRoles(VARIANT* roles);

Interface IConverter

Last Modified:

3-Jan-2012 Remove superseded

3-May-2011 Superseded

3-Feb-2009 New topic

GetReferenceMarker Method

Description Returns a handle to the reference marker.

VB Syntax *meas*.**GetReferenceMarker**

Variable [\(Type\)](#) - Description

meas A Measurement (**object**)

Return Type Object

Default Not Applicable

Examples `meas.GetReferenceMarker`

C++ Syntax HRESULT GetReferenceMarker(IMarker**
refMarker)

Interface IMeasurement

GetRequiredEtermNames

Description Returns an array of strings specifying the error terms required by the caltype's correction algorithm in order to correct the specified parameter.

This function interrogates a specific caltype (caltypeGUID) for the list of error terms it would need in order to correct the specified parameter. All the standard S Parameter calibration types embed port specifiers in the error term name. The specific port information is gleaned from the passed parameter.

For example, to query the error term requirements specific to a two port cal on ports 1 and 3, issue this with a parameter of S13 or S31. The buffer names returned will be formatted in this way:

Full 1 Port SOLT(1,3):TransmissionTracking(3,1)

VB Syntax **EtermNames = GetRequiredEtermNames**(CalTypeGUID As String, Parameter As String)

Variable **(Type) - Description**

caltypeGUID: [in] the GUID of the desired calibration type

parameter [in] string specifying the parameter to be corrected

EtermNames [out] array of strings containing the error term names.

Note: In C++ Allocated by server. Must be freed by caller using SysFreeString.

Return Type Not Applicable

Default Not Applicable

Examples enames = GetRequiredEtermNames(ctGUID, Parm)

C++ Syntax HRESULT GetRequiredEtermNames(BSTR caltypeGUID, BSTR parameter, VARIANT* EtermNames)

Interface ICalManager2

GetScalar Method

Description Retrieves scalar data (ONE number per data point) from the specified location.

Note: This method exists on a non-default interface. If you cannot access this method, use the [Get Data](#) Method on IMeasurement.

VB Syntax `measData.getScalar location, format, numPts, data`

Variable (Type) - Description

measData An IArrayTransfer interface which supports the Measurement object

location (**enum NADataStore**) - Where the data you want is residing. Choose from:

0 - naRawData

1 - naCorrectedData

2 - naMeasResult

3 - naRawMemory

4 - naMemoryResult

5 - naDivisor - When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using [DataToDivisor Method](#).

format (**enum NADataFormat**) - Format in which you would like the data. Choose from:

0 - naDataFormat_LinMag

1 - naDataFormat_LogMag

2 - naDataFormat_Phase

3 - naDataFormat_Polar

4 - naDataFormat_Smith

5 - naDataFormat_Delay

6 - naDataFormat_Real

7 - naDataFormat_Imaginary

- 8 - naDataFormat_SWR
- 9 - naDataFormat_PhaseUnwrapped
- 10 - naDataFormat_InverseSmith
- 11 - naDataFormat_Kelvin
- 12 - naDataFormat_Fahrenheit
- 13 - naDataFormat_Centigrade

Note: Polar, Smith, and Inverse Smith are invalid formats for this command. See [Get Complex Method](#).

[Learn more about Data Format](#).

numPts (**long integer**) - Number of data points requested
 [out] - specifies number of data elements returned
 [in] - specifies the data being requested or the capacity of the *dScalar* array

data (**single**) - Array to store the scalar data.

Return Type Single

Default Not Applicable

Examples

```
Dim dScalar() As Single
Dim measData As IArrayTransfer
Set measData = app.ActiveMeasurement
Dim numpts as Long
numpts = app.ActiveChannel.NumberOfPoints
ReDim dScalar(numPoints)

measData.getScalar naCorrectedData, naDataFormat_LogMag, numpts,
dScalar(0)
Print dScalar(0), dScalar(1)
```

C++ Syntax HRESULT getScalar(tagNADataStore DataStore, tagNADataFormat DataFormat, long* pNumValues, float* pVals)

Interface IArrayTransfer

Last Modified:

1-Oct-2007 Added temperature formats

GetShortcut Method

Description From the index (line number in the user interface) returns the Title, Path, and optional argument strings, of the specified Macro (shortcut). Use this method to list the titles and paths of macros in the analyzer.

VB Syntax *app.GetShortcut index, title, path, arguments*

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

index (**long**) - Number of the macro. Use a number between **1** and **25**.

title (**string**) - **Title** of the specified macro. (Appears in the softkey label)

path (**string**) - Pathname of the specified macro.

arguments (**string**) - Arguments for the specified macro

Return Type String

Default Not Applicable

Example

```
Dim t As String
Dim p As String
Dim arg As String
Dim i As Integer
For i = 1 to 25
    app.GetShortcut i,t,p,arg
    Print t,p
Next
```

C++ Syntax HRESULT GetShortcut(long Number, BSTR* title, BSTR* pathname, BSTR* arguments)

Interface IApplication

Remarks Shortcuts can also be defined and accessed using the macro key on the front panel. However, the benefit of this feature is primarily for the interactive user

Last Modified:

29-Apr-2013 Updated to 25 total

GetSnPData Method **Superseded**

Description	Note: this command is replaced by Get SnpDataWithSpecifiedPorts Method . Reads SnP data from the selected measurement. Learn more about SnP that is returned from the PNA.
VB Syntax	<i>data = meas.GetSnPData type</i>
Variable	(Type) - Description
<i>data</i>	Variant array to store the data.
<i>meas</i>	A Measurement (object)
<i>type</i>	(string) - Type of SnP data to return. If unspecified, <n> is set to 2. Choose from: "S1P" returns 1-Port data for the active measurement if the active measurement is a reflection parameter such as S11 or S22. The behavior is UNDEFINED if the active measurement is a transmission parameter such as an S21. "S2P" returns data for the current 2-port measurement (4 S-parameters). "S3P" returns data for the current 3 port measurement (9 S-parameters). Valid only on instruments with 3 ports or more. "S4P" returns data for the current 4 port measurement (16 S-parameters). Valid only on instruments with 4 ports or more. SnP data can be output using several data formatting options. See SnPFormat Property
Return Type	Variant - 3 dimensional array. <ul style="list-style-type: none">• First dimension size is number of parameters returned.• Second dimension size is number of points in the channel• Third dimension size is 2 (real,imaginary)

For example:

Data(0,5,1) returns the imaginary value of the fifth data point of S11 (if the s2p

request includes port #1)

Default Not Applicable

Examples `snp = meas.GetSnPData("s1p")`

C++ Syntax HRESULT GetSnPData(BSTR snptype, VARIANT * response)

Interface IMeasurement3

Last Modified:

27-Sep-2012 Added return 1-port detail

14-Nov-2008 Added returned data detail

GetSnPDataWithSpecifiedPorts Method

Description	<p>Note: This command replaces Get SnPData. This command is more explicit regarding the data to be returned, and works for PNAs with multiport test sets.</p> <p>Reads SnP data for the measurement by specifying the PNA port numbers. Learn more about SnP that is returned from the PNA.</p>
VB Syntax	<code>data = meas.GetSnPDataWithSpecifiedPorts ports</code>
Variable	(Type) - Description
<i>data</i>	(Variant) array to store the data.
<i>meas</i>	A Measurement (object)
<i>ports</i>	(Variant Array) One-dimensional array containing the list of port numbers for which data is required.
Return Type	Variant - 3 dimensional array.
	<ul style="list-style-type: none">• First dimension size is number of parameters returned.• Second dimension size is number of points in the channel• Third dimension size is 2; format of the data is specified with SnPFormat Property.

For example:

Data(0,5,1) returns the imaginary part of the fifth data point of S11 (if the s2p request includes port #1)

Default Not Applicable

Example `'This VBScript example can be pasted into a notepad file and run on the PNA as a macro. Learn how.`

```
Dim pna
Dim meas
Dim param
Dim point
Dim snp
Dim ports

'List the port numbers for required data
```

```

ports = Array(3,4)

Set pna = CreateObject("AgilentPna835x.application")

Set meas = pna.ActiveMeasurement

'limit amount of data to display

set chan=pna.ActiveChannel

chan.NumberOfPoints=2

snp = meas.GetSnpDataWithSpecifiedPorts (ports)

' returns a 3 dimensional array

' snp(param,point,data pair)

'-----

' show me the data

For param = LBound(snp, 1) To UBound(snp, 1)

  MsgBox ("Parameter: " & (param + 1))

  For point = LBound(snp, 2) To UBound(snp, 2)

    MsgBox "Point:" & (point + 1) & " " & snp(param, point, 0) &
    "," & snp(param, point, 1)

  Next

Next

```

C++ Syntax HRESULT GetSnpDataWithSpecifiedPorts(VARIANT portsToMeasure,VARIANT* response);

Interface IMeasurement7

Last modified:

13-Jun-2011 Fixed data format
13-Nov-2008 Added detail to return type
9/18/06 MQ Added for multiport

getSourcePowerCalDataEx Method

Description **Note:** This method replaces [getSourcePowerCalData Method](#)

Retrieves (as variant data type) source power calibration data, if it exists, from the channel.

If the channel is sweeping the source backwards, then the first data point is the highest frequency value; the last data point is the lowest. Use the [Get X-Axis Values](#) command to return the X-axis values in the displayed order.

Note: This method returns a variant which is less efficient than methods available on the ISourcePowerCalData interface

VB Syntax `data = chan.getSourcePowerCalDataEx (buffer, srcPort)`

Variable **(Type) - Description**

data **(variant)** – Array to store the data.

chan **(object)** – A [Channel](#) object

buffer **(enum NASourcePowerCalBuffer)** - The requested source power cal data buffer.

0 - naCorrectionValues Last iteration of Cal data

1 - naPriorIterationCorrectionValues Prior iteration of Cal data. This argument can be used to determine the final power reading at each point of the power cal, for a cal that did not pass tolerance limits.

The following formula can be used to determine the power reading (in dB):

Power reading = Target power at the source port + specified power cal offset value + 'prior' iteration corr value – actual power corr value.

The "actual" value in this equation is returned with **naCorrectionValues**.

srcPort **(long integer)** – The source port for which calibration data is being requested.

Note: If the port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-x model, then you must use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#).

Return Type Variant array – automatically dimensioned to the size of the data.

Default Not Applicable

Examples

```
Dim varData As Variant
Const port1 As Long = 1
varData = chan.getSourcePowerCalDataEx (naCorrectionValues,
port1)
'Print the data
For i = 0 to chan.NumberOfPoints - 1
Print varData(i)
Next i
```

C++ Syntax

HRESULT getSourcePowerCalDataEx(tagNASourcePowerCalBuffer
bufSelect, long sourcePort, VARIANT *pData);

Interface IChannel4

Last Modified:

March 17, 2010	Added Prior argument
24-Apr-2008	Added note for string names
27-Jun-2007	Updated for PNA-X source port names

getSourcePowerCalDataScalarEx Method

Description	<p>Note: This method replaces getSourcePowerCalDataScalar Method</p> <p>Retrieves (as scalar values) source power calibration data, if it exists, from this channel.</p> <p>If the channel is sweeping the source backwards, then the first data point is the highest frequency value; the last data point is the lowest. Use the Get X-Axis Values2 command to return the X-axis values in the displayed order.</p> <p>Note: This method exists on a non-default interface. If you cannot access this method, use the getSourcePowerCalDataEx Method on IChannel4.</p>
VB Syntax	<i>chanData</i> . getSourcePowerCalDataScalarEx <i>buffer</i> , <i>srcPort</i> , <i>numValues</i> , <i>data</i>
Variable	(Type) - Description
<i>chanData</i>	(interface) – An ISourcePowerCalData2 interface on the Channel object.
<i>buffer</i>	(enum NASourcePowerCalBuffer) - The requested source power cal data buffer.
	<p>0 - naCorrectionValues Last iteration of Cal data.</p> <p>1 - naPriorIterationCorrectionValues Prior iteration of Cal data. This argument can be used to determine the final power reading at each point of the power cal, for a cal that did not pass tolerance limits.</p> <p>The following formula can be used to determine the power reading (in dB):</p> <p>Power reading = Target power at the source port + specified power cal offset value + 'prior' iteration corr value – actual power corr value.</p> <p>The "actual" value in this equation is returned with naCorrectionValues.</p>
<i>srcPort</i>	(long integer) – The source port for which calibration data is being requested.
	<p>Note: If the port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-x model, then you must use chan.getPortNumber to translate the string into a port number. To learn more see Remotely Specifying a Source Port.</p>
<i>numValues</i>	(long integer) – Number of data values.

[out] – specifies number of data values returned.

[in] – specifies number of values being requested (this must not be larger than the capacity of the data array).

data (**single**) – Array to store the data.

Return Type Single

Default Not Applicable

Examples

```
Dim numValues As Long
Dim scalarCalValues() As Single
Dim chanData As ISourcePowerCalData2
Const port1 As Long = 1
numValues = app.ActiveChannel.NumberOfPoints
ReDim scalarCalValues(numValues)
Set chanData = app.ActiveChannel

chanData.getSourcePowerCalDataScalarEx naCorrectionValues,
port1, numValues, scalarCalValues(0)

'Print the data
For i = 0 to numValues - 1
Print scalarCalValues(i)
Next I
```

C++ Syntax HRESULT getSourcePowerCalDataScalarEx(tagNASourcePowerCalBuffer bufSelect, long sourcePort, long *pNumValues, float *pData);

Interface ISourcePowerCalData2

Last Modified:

17-Mar-2010	Added Prior argument
24-Apr-2008	Added note for string names
27-Jun-2007	Updated for PNA-X source port names

GetStandard Method **Superseded**

Description This command has been replaced with [Get StandardByString](#)

Returns standard acquisition data from the Cal Set. The returned data is complex pairs.

Learn more about [Reading and Writing Cal Data](#)

See examples of [Reading](#) and [Writing](#) Cal Set Data

VB Syntax `data = CalSet.getStandard (standard, rcv, src)`

Variable [\(Type\)](#) - Description

data **(Variant)** Two-dimensional safe array to store the returned data. Memory for the returned Variant is allocated by the PNA and must be released by client.

Note: See also [getStandardComplex](#) on the ICalData2 interface to avoid using the variant data type.

CalSet **A Cal Set (object)**

standard **(enum NACalClass)** Standard data to be read. Choose from:

- 1 - naClassA
- 2 - naClassB
- 3 - naClassC
- 4 - naClassD
- 5 - naClassE
- 6 - naReferenceRatioLine
- 7 - naReferenceRatioThru

SOLT Standards

- 1 - naSOLT_Open
- 2 - naSOLT_Short
- 3 - naSOLT_Load
- 4 - naSOLT_Thru
- 5 - naSOLT_Isolation

TRL Standards

- 1 - naTRL_Reflection
- 2 - naTRL_Line_Reflection

3 - naTRL_Line_Tracking

4 - naTRL_Thru

5 - naTRL_Isolation

rcv (long) - Receiver Port

src (long) - Source Port

Return Type (variant)

Default Not Applicable

Examples

```
Dim varStd As Variant
Dim varStd2 As Variant

Cal Set.OpenCalSet( naCalType_TwoPortSOLT, 1, 2)
varStd = CalSet.getStandard(naSOLT_Thru,2,1)
varStd2 = Cal Set.getStandard(naSOLT_Thru,1,2)
Cal Set.CloseCalSet( )
```

C++ Syntax HRESULT getStandard(tagNACalClass stdclass, long ReceivePort, long SourcePort, VARIANT* pData)

Interface ICalSet

GetStandardByString Method

Description Returns standard acquisition data from the Cal Set. The returned data is complex pairs.

Learn more about [Reading and Writing Cal Data](#)

See examples of [Reading](#) and [Writing](#) Cal Set Data

VB Syntax `data = calSet.GetStandardByString(stdName)`

Variable [\(Type\) - Description](#)

data **(Variant)** Two-dimensional safe array to store the returned data. Memory for the returned Variant is allocated by the PNA and must be released by client.

Note: See also [Get StandardComplexByString](#) on the ICalData2 interface to avoid using the variant data type.

calSet A [CalSet \(Object\)](#)

stdName **(String)** The string used to identify a particular standard in the Cal Set. An example string requesting the data for the Load standard in a full 2 port cal might be "S11C(3,3)".

Return Type Variant

Default Not Applicable

Examples See an [example](#)

C++ Syntax HRESULT GetStandardByString(BSTR bufferName, VARIANT* pdata)

Interface ICalSet2

GetStandardComplex Method **Superseded**

Description This command is [replaced](#) with [Get StandardComplexByString](#)

Returns standard acquisition data from the Cal Set. The returned data is complex pairs.

Learn more about [Reading and Writing Cal Data](#)

See examples of [Reading](#) and [Writing](#) Cal Set Data

Note: This method exists on a non-default interface. If you cannot access this method, use the [GetStandard](#) Method on ICal Set

VB Syntax `ICalData2.getStandardComplex class, rcv, src, numPts, real(), imag()`

Variable (Type) - Description

ICalData2 An [ICalData2](#) pointer to the Cal Set object

class (**enum NACalClass**) Standard data to be read. Choose from:

- 1 - naClassA
- 2 - naClassB
- 3 - naClassC
- 4 - naClassD
- 5 - naClassE
- 6 - naReferenceRatioLine
- 7 - naReferenceRatioThru

SOLT Standards

- 1 - naSOLT_Open
- 2 - naSOLT_Short
- 3 - naSOLT_Load
- 4 - naSOLT_Thru
- 5 - naSOLT_Isolation

TRL Standards

- 1 - naTRL_Reflection
- 2 - naTRL_Line_Reflection
- 3 - naTRL_Line_Tracking
- 4 - naTRL_Thru

5 - naTRL_Isolation

rcv (**long**) - Receiver Port

src (**long**) - Source Port

numPts (**Long**) An In/Out parameter.

On the way **in**, you specify the **max** number of values being requested.

On the way **out**, the PNA returns number of values actually returned.

real() (**single**) - array to accept the real part of the calibration data. One-dimensional for the number of data points.

imag() (**single**) - array to accept the imaginary part of the calibration data. One-dimensional for the number of data points.

Return Type (**single**)

Default Not Applicable

Examples

```
Dim numpts as long
numpts = ActiveChannel.NumberOfPoints
ReDim r(numpts) ' real part
ReDim i(numpts) ' imaginary part
Dim Cal Set as Cal Set
set Cal Set = pna.GetCalManager.GetCal SetByGUID( txtGUID )
Dim sData As ICalData2
Set sData = Cal Set
sdata.getStandardComplex naSOLT_Open, 1, 1, numpts, r(0), i(0)
```

C++ Syntax HRESULT getStandardComplex(tagNACalClass stdclass, long ReceivePort, long SourcePort, long* pNumValues, float* pReal, float* pImag)

Interface ICalData2

GetStandardComplexByString Method

Description	Returns standard acquisition data from the Cal Set. Learn more about Reading and Writing Cal Data See examples of Reading and Writing Cal Set Data
VB Syntax	<i>ICalData3</i> . GetStandardComplexByString <i>stdName</i> , <i>InumPoints</i> , <i>real(0)</i> , <i>imag(0)</i>
Variable	(Type) - Description
<i>ICalData3</i>	An ICalData3 pointer to a CalSet (Object)
<i>stdName</i>	(String) The string used to identify a particular standard in the Cal Set. An example string requesting the data for the Load standard in a full 2 port cal might be "S11C(3,3)".
<i>InumPoints</i>	(Long) An In/Out parameter. On the way in , you specify the max number of values being requested. On the way out , the PNA returns number of values actually returned.
<i>real</i>	(Single) The real component of the complex data.
<i>imag</i>	(Single) The imaginary component of the complex data.
Return Value	Single
Default	Not Applicable
Examples	See example
C++ Syntax	HRESULT GetStandardComplexByString(BSTR bufferName, long* InumPoints, float* real, float* imag);
Interface	ICalData3

GetStandardsList Method **Superseded**

Description **Note:** This command is replaced by [CalSet.getStandardList2](#).

Returns the list of Standards contained in this Cal Set for the CalType specified in the [OpenCal Set](#) method. Learn more about [reading and writing Cal Data using COM](#).

The list is a comma separated, textual representation of the error terms with the term name followed by the port path in parentheses.

Standard (n, n),
Standard (m, n)

Before calling this method you must open the Cal Set with [OpenCal Set](#). If the Cal Set is not open, this method returns E_NA_Cal Set_ACCESS_DENIED.

Use [StringToNACalClass](#) to convert the list entrees to values that can be used with [GetStandard](#) and [PutStandard](#).

Note: The port path designation (m n) indicates the receive and source ports for the measurement. Shorts, opens and loads are single port devices, designated in this list by (n n) where n equals the port to which the device is connected. These devices are all characterized by reflection measurements. The dual port thru device is characterized by both transmission and reflection measurements in order to compensate for load match and tracking terms. The notation (n n) indicates the reflection measurement for this device. The notation (m n) indicates the transmission measurement, where the source and receive ports are different.

VB Syntax *CalSet*.**GetStandardsList** (*count*, *list*)

Variable [\(Type\)](#) - Description

CalSet **(object)** - A Cal Set object

count **(long [out])** - indicates the number of items returned in the list

list **(string)** - Variable to store the returned Comma separated list of items.

Return Type String

Default Not Applicable

Examples

```
dim count as Integer
dim list as string
OpenCalSet (naCalType_TwoPortSOLT, 1, 2)
GetStandardsList( count, list)
CloseCalSet( )
```

Assuming the Cal Set contained the full set of standards for this two port cal, the returned list would be:

```
"Open(1 1),
Short(1 1),
Load(1 1),
Thru(1 1),
Isolation(2 1),
Open(2 2),
Short(2 2),
Load(2 2),
Thru(2 2),
Isolation(1 2)
Thru(2 1),
Thru(1 2)"
```

C++ Syntax HRESULT GetStandardsList(long* count, BSTR* list);

Interface ICalSet

GetStandardList2 Method

Description	Returns a list of standards contained by this Cal Set for the specified Cal Type.
VB Syntax	<i>list</i> = <i>calset</i> . GetStandardList2 (<i>calType</i>)
Variable	(Type) - Description
<i>calset</i>	(object) - A CalSet object
<i>list</i>	(Variant) Variant containing a string array of standards for the specified calType.
<i>calType</i>	(String) The string used to identify Cal Set data as belonging to a specific Cal Type. This string is used as a filter so that only the standard names of interest are returned. If the prefix is empty, all names are returned. An example prefix for a two port cal on ports 2 and 3 might be: "Full 2 Port Cal (2,3)".
Return Type	Variant
Default	Not Applicable
Examples	See an example
C++ Syntax	HRESULT GetStandardList2 (BSTR caltype, VARIANT* list)
Interface	ICalSet2

GetStandardsForClass Method

Description Get the calibration standard numbers for a specified calibration class. To set the calibration number use [SetStandardsForClass Method](#)

VB Syntax *calkit*.**GetStandardsForClass** (*calclassorder*, *std1*, *std2*, *std3*, *std4*, *std5*, *std6*, *std7*)

Variable [\(Type\)](#) - Description

calKit A CalKit (**object**)

calclassorder (**enum NA CalClassOrder**) Choose from:

- 0 - naRefl_1_S11
- 1 - naRefl_2_S11
- 2 - naRefl_3_S11
- 3 - naTran_1_S21
- 4 - naRefl_1_S22
- 5 - naRefl_2_S22
- 6 - naRefl_3_S22
- 7 - naTran_1_S12
- 8 - naRefl_1_S33
- 9 - naRefl_2_S33
- 10 - naRefl_3_S33
- 11 - naTran_1_S32
- 12 - naTran_1_S23
- 13 - naTran_1_S31
- 14 - naTran_1_S13
- 15 - naTRL_T
- 16 - naTRL_R
- 17 - naTRL_L

std1...std7 (**long**) Calibration Standard Number. Nominal values from **1** through **30**. **0** indicates that a standard number has not been selected.

Return Type Not applicable

Default Not applicable

Examples `calkit.GetStandardsForClass naRef1_3_S11, std1, std2, std3, std4, std5, std6, std7`

C++ Syntax HRESULT GetStandardsForClass(NACalClassOrder calclassorder, long std1, long std2, long std3, long std4, long std5, long std6, long std7)

Interface ICalKit

Read-only

GetStepDescription Method

Description	Returns the description of the specified step in the calibration process. For an ECal User Characterization this returns the description of the specified step in the ECal User Characterization process.
VB Syntax	<i>value</i> = <i>obj</i> . GetStepDescription (<i>n</i>)
Variable	(Type) - Description
<i>value</i>	(string) - Variable to store the returned number of steps.
<i>obj</i>	Any of the following: GuidedCalibration (object) SMCType (object) VMCType (object) ECalUserCharacterizer (object)
<i>n</i>	(Long) Step in the process. Use GenerateSteps to determine the total number of steps.
Return Type	String
Default	Not Applicable
Examples	<pre>value = SMC.GetStepDescription(5)</pre>
C++ Syntax	HRESULT get_GetStepDescription(long step, BSTR* str);
Interface	IGuidedCalibration SMCType VMCType IECalUserCharacterizer

Last Modified:

4-Nov-2008 Added ECal object

GetSupportedALCModes Method

Description Returns the valid ALC Modes for the PNA.

See [ALCLevelingMode](#) for a list of supported ALC Modes.

VB Syntax *value* = *chan*.**GetSupportedALCModes** (*sourcePort*)

Variable [\(Type\)](#) - Description

value Variant Array variable to store the returned valid ALC Modes.

chan **(object)** - A [Channel](#) object

sourcePort **(long integer)** - Source port.

Note: If the port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-x model, then you must use [chan.getPortNumber](#) to translate the string into a port number. To learn more see [Remotely Specifying a Source Port](#).

Return Type Variant array

Default Not Applicable

Examples

```
modes = chan.GetSupportedALCModes(4) 'Read
```

C++ Syntax HRESULT GetSupportedALCModes(long port, VARIANT * ALCModes);

Interface IChannel9

Last Modified:

22-Apr-2010	Fixed C++syntax and interface (9)
24-Apr-2008	Added note for string names
30-Apr-2007	Edited for src strings
16-Jan-2007	MX New topic

GetTestResult Method

Description Returns the result of limit line testing. There are three ways to use this command:

- If neither optional parameter is specified, limit results for ALL data is returned.
- If one parameter is specified (*start*), the limit result for that data point is returned.
- If both parameters are specified, limit results are returned beginning with *start*, and ending with (*start+size*)-1

Note: In 'strongly-typed' languages such as C#, all parameters must be specified.

VB Syntax `testRes = limts.GetTestResult [start,size]`

Variable [\(Type\) - Description](#)

testRes **(enum NALimitTestResult)** - A dimensioned variable to store test results. If a limit line is not tested, a PASS is returned.

0 - naLimitTestResult_None

1 - naLimitTestResult_Fail

2 - naLimitTestResult_Pass

limts A LimitTest (**object**)

start (**long**) - Optional argument. A start data point number to return limit test results.

size (**long**) - Optional argument. Number of data points from *start* to return limit test results.

Return Type Long Integer

Default Not Applicable

Examples

```
Dim testRes As NALimitTestResult
testRes = limts.GetTestResult
Select Case testRes

Case 1
Print "Fails"

Case 2
Print "Pass"

End Select
```

C++ Syntax HRESULT GetTestResult(long lStart, long lSize, tagNALimitTestResult *pVal)

Interface ILimitTest

Last modified:

12-Sept-2011 Added C# note

GetTraceStatistics Method

Description Returns all four Trace Statistics. To retrieve individual Trace statistics, use [Mean](#), [PeakToPeak](#), [StandardDeviation](#) properties. Use [ShowStatistics](#) to display the statistics of the screen.

VB Syntax *meas.GetTraceStatistics pp,mean,stdev*

Variable [\(Type\)](#) - Description

meas A Measurement (**object**)

pp,mean,stdev (**double**) - Dimensioned variables to store the returned values

Return Type Double

Default Not Applicable

Examples

```
'Dimension variables
Dim pp As Double
Dim mean As Double
Dim stdv As Double
meas.GetTraceStatistics pp, mean, stdv
```

C++ Syntax HRESULT GetTraceStatistics(double* pp, double* mean, double* stdDeviation)

Interface IMeasurement

GetXAxisValues2 Method

Description Returns the channel's X-axis values into a dimensioned Typed array. GetXAxisValues2 is a convenient method for determining the frequency of each point when the points are not linearly spaced - as in segment sweep.

Note: This method will fail if called using a scripting client such as VBScript or Agilent Vee, ([see remarks](#))

Note: In Segment Sweep, chan.[NumberOfPoints](#) will return the total number of data points for the combined segments.

VB Syntax *chan*.GetXAxisValues2 *numPts*,*data*

Variable [\(Type\)](#) - Description

chan **(object)** - A Channel object

numPts **(long integer)** - Number of data points in the channel

data **(double)** Single dimensioned array of data matching the number of points in the channel.

Return Type double

Default Not applicable

Examples

```
Dim App As Application
Set App = New Application
Dim numPoints As Long
Dim values() As Double
numPoints = App.ActiveChannel.NumberOfPoints
ReDim values(numPoints)
App.ActiveChannel.GetXAxisValues2 numPoints, values(0)
Print values(0), values(1)
```

C++ Syntax HRESULT GetXAxisValues2(long* pNumValues, double* stimulus)

Interface IChannel

Remarks:

This method will fail if called using a scripting client such as VBScript or Agilent Vee. Use the [GetXAxisValues](#) method as a replacement for these COM environments.

This method also cannot be called using late-bound typing in Visual Basic. For instance, if, in the

example above, the first line were replaced with "Dim App as Object", then this method would fail.

GetXAxisValues Method

Description Returns the stimulus values for the measurement. To understand how this property is useful, see [IMeasurement2 Interface](#).

VB Syntax *data* = *meas*.**GetXAxisValues**

Variable [\(Type\) - Description](#)

data **(Variant)** Array to store the data.

meas A Measurement **(object)**

Return Type Variant

Default Not Applicable

Examples

```
Dim varData As Variant
Dim i As Integer
varData = meas.GetXAxisValues
'Print Data
For i = 0 To meas.NumberOfPoints - 1
Print varData(i)
Next i
```

[See C++ example](#)

C++ Syntax HRESULT GetXAxisValues(VARIANT* xData);

Interface IMeasurement2

Last Modified:

15-Oct-2007 Added link to C++ example

GetXAxisValues Method

Description Returns the channel's X-axis values. GetXAxisValues is a convenient method for determining the frequency of each point when the points are not linearly spaced - as in segment sweep.

See the [Measurement2 Interface](#) to learn how this method differs from [meas.GetXAxisValues](#).

Note: This method returns a variant which is less efficient than [GetXAxisValues2](#).

Note: In Segment Sweep, chan.NumberofPoints will return the total number of data points for the combined segments.

VB Syntax `data = chan.GetXAxisValues`

Variable [\(Type\)](#) - Description

data Variant array to store the data.

chan A Channel (**object**)

Return Type Variant

Default Not Applicable

Examples

```
Dim varData As Variant
Dim i As Integer
varData = chan.GetXAxisValues
'Print Data
For i = 0 To chan.NumberOfPoints - 1
    Print varData(i)
Next i
```

C++ Syntax HRESULT GetXAxisValues (VARIANT* xData)

Interface IChannel

HasCalType Method

Description Verifies that the Cal Set object contains the error terms required to perform the specified correction (CalType) to an appropriate measurement.

The argument list includes specifiers for up to 3 ports. The number of arguments required depends on the CalType specified. The value for each port is set to 0 if not specified.

VB Syntax *check* = *CalSet*.**HasCalType** (*calType*, *p1*, *p2*, *p3*)

Variable [\(Type\) - Description](#)

check **(boolean)** - variable to store the returned value

TRUE (1) - Cal Set has all of the error terms necessary to apply the specified correction CalType.

FALSE(0) - Cal Set DOES NOT have all of the error terms necessary to apply the specified CalType.

CalSet **(object)** - A Cal Set object

calType (enum as naCalType) - type of correction to be applied. Choose from:

Caltype	p arguments required
0 - naCalType_Response_Open	p1
1 - naCalType_Response_Short	p1
2 - *naCalType_Response_Thru	p1 (rcv), p2 (src)
3 - *naCalType_Response_Thru_And_Isol	p1 (rcv), p2 (src)
4 - naCalType_OnePort	p1
5 - naCalType_TwoPort_SOLT	p1, p2
6 - naCalType_TwoPort_TRL	p1, p2
7 - naCalType_None	N/A
8 - naCalType_ThreePort_SOLT	p1, p2, p3
9 - Custom	N/A
10 - naCalType_FourPort_SOLT	p1, p2, p3

(port 4 is assumed)

* order of port arguments is significant for these CalTypes

p1 **(long)** - required. This argument must be specified.

This specifies either:

- the one significant port for an open/short response cal or a 1 port cal.
- or one of the ports involved in a 2, 3, or 4 port cal
- or the **receive** port for a thru response / thru-isolation cal.

*p*² **(long)** - required for any CalType involving more than one port

This specifies either:

- one of the ports involved in a 2, 3, or 4 port cal (order independent)
- or the **source** port for a thru response / thru-isolation cal

*p*³ **(long)** - required for 3 and 4-port cal

This specifies one of the ports involved in a 3 or 4 port cal (order independent)

Return Type VARIANT_BOOL

Default Not Applicable

Examples `value = CalSet.HasCalType(naCalType_TwoPort_TRL, 1, 2)`

C++ Syntax HRESULT HasCalType(tagNACalType, long port1, long port2, long port3, BOOL *pVal);

Interface ICalSet

Hold Method

Description Puts the channel in Hold - not sweeping.

See [chans.Hold](#) to put ALL channels in hold.

VB Syntax *chan.Hold* [*sync*]

Variable [\(Type\)](#) - Description

chan A [Channel](#) (**object**)

[*sync*] The [*sync*] argument is ignored.

Program control ALWAYS waits until the channel is in the Hold state.

Return Type Not Applicable

Default Not Applicable

Examples `chan.Hold`

C++ Syntax HRESULT Hold

Interface IChannel

Last Modified:

20-Apr-2010 Updated sync

Hold (channels) Method

Description	Places ALL channels in hold mode. To resume all channels sweeping, use chans.Resume . (Must be the same instance of chans). To place a single channel in hold mode, use channel.Hold Method.
VB Syntax	<i>chans</i> . Hold
Variable	(Type) - Description
<i>chans</i>	A Channel collection (object)
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>chans.Hold</code>
C++ Syntax	HRESULT Hold();
Interface	IChannels2

Description Imports the Guided Power Cal Set (from an existing SMC Cal) or Phase Reference Cal into the current SMC calibrations.

For the Guided Power Cal:

The port of the mixer input must have the same source attenuator setting between the SMC channel and the Guided Power Cal Set. The frequencies of the Guided Power Cal must include all the mixer frequencies. Interpolation will be applied to the Guided Power Cal frequencies if they do not exactly match.

For the Phase Reference Cal:

The port of the mixer input must have the same source attenuator setting as used in the phase reference cal. The phase reference cal must include all the mixer frequencies. Interpolation will be applied to the phase reference cal frequencies if they do not exactly match. [Learn more about Phase Reference Cal.](#)

The following error message may appear (it is not written to the PNA Error Log):

Interpolation target is out of range. Cannot interpolate when incompatible frequency ranges occur.

VB Syntax `smc.ImportDataSet (calset, dataName)`

Variable [\(Type\)](#) - Description

`smc` [SMCType](#) (object)

`calset` (String) Name of existing SMC Cal Set from which cal data is imported.

`dataName` (String) Name of the data set. Choose from:

- "POWER_STEP" - Import Guided Power Cal data.
- "POWER_AND_PHASE" - Import the Phase Reference + power cal data. When this command is sent, the SMC Cal Method is automatically set to Use Phase Reference Cal Set. [Learn more.](#) There is no other command to set this.

Return Type Not Applicable

Default Not Applicable

Examples `smc.ImportDataSet("MySMCCal", "POWER_STEP")`

[See example program](#)

C++ Syntax HRESULT ImportDataSet(BSTR csName, BSTR dataset);

Interface SMCType4

Last Modified:

24-Feb-2012 Edit for Phase Reference

7-Mar-2011 Edited for interpolation

22-Dec-2009 MX New topic

ImportLibrary Method

Description	Imports an Equation Editor DLL.
VB Syntax	<i>equation</i> . ImportLibrary <i>location</i>
Variable	(Type) - Description
<i>equation</i>	A MeasurementEquation object
<i>location</i>	(string) – Full path and filename of the *.dll to be imported.
Return Type	Not Applicable
Default	Not Applicable
Examples	<pre>equation.ImportLibrary "C:/Program Files/Agilent/Network Analyzer/UserFunctions/Expansion.dll"</pre>
C++ Syntax	HRESULT ImportLibrary(BSTR filename);
Interface	IMeasurementEquation2

Last Modified:

10-Jan-2011 New topic

Initialize Method

Description Begins a calibration.

Note: *chan* must be the active channel.

For ECal User Characterization, use [Initialize \(ECal\)](#).

VB Syntax *obj*.**Initialize** (*chan*, *useCalStorPref*)

Variable [\(Type\) - Description](#)

obj Any of the following:

[GuidedCalibration](#) (object)

[SMCType](#) (object)

[VMCType](#) (object)

chan (Long) Channel number to calibrate.

useCalStorPref (boolean)

True or **1** - Assignment of Cal Set will be based on the setting of the [RemoteCalStoragePreference](#) COM property.

False or **0** – If the channel currently has a selected Cal Set, the calibration will be stored to that Cal Set. Otherwise, the assignment of Cal Set is based upon the setting of the [RemoteCalStoragePreference](#) COM property.

Return Type Not Applicable

Default Not Applicable

Examples `smc.Initialize(2, True)`

C++ Syntax HRESULT put_Initialize(long channelnumber, VARIANT_BOOL bCalPref);

Interface IGuidedCalibration

SMCType

VMCType

Last Modified:

4-Nov-2008 Added Note about ECal User Char

InitializeEx Method

Description **Note:** This property replaces [Initialize \(ECal\) Method](#).

Initiates a User Characterization of an ECal module. The specified channel number must be an S-parameter measurement channel. The channel must already be calibrated using the same, or greater number of PNA ports as the ECal module. Also, the calibrated PNA ports must begin with Port 1 and use sequential port numbers.

For characterizations that are to be saved to the ECal module, the User Characterization number must already be set before issuing this command using [CharacterizationNumber Property](#). For characterizations that are to be saved to the PNA disk memory, setting the User Characterization number is not necessary.

After this command is executed, subsequent commands can be used to query the number of measurement steps, issue the acquisition commands, query the connection description strings, and subsequently complete an Ecal User characterization.

VB Syntax *userChar*.**InitializeEx** (*chan*,*bool*)

Variable [\(Type\) - Description](#)

userChar [ECalUserCharacterizer](#) (object)

chan (Long) Channel number of a calibrated S-parameter channel.

bool (Boolean) Choose from:

True Check ECal memory to ensure that a new characterization with the channel's current number of points will fit in the module memory. Select for User Characterizations to be stored in internal ECal memory.

False Skip the check. Select for User Characterizations is to be stored to PNA disk memory.

Return Type Not Applicable

Default Not Applicable

Examples `userchar.InitializeEx(2,True)`

C++ Syntax HRESULT put_InitializeEx(long chanNum,VARIANT_BOOL bCheck);

Interface IECalUserCharacterizer2

Last Modified:

25-Aug-2009 MX New topic

Initialize Method Superseded

Description	Note: This command is replaced with InitializeEx Method Initiates a User Characterization of an ECal module. The specified channel number must be an S-parameter measurement channel. The User characterization number must already be set before issuing this command using CharacterizationNumber Property .
VB Syntax	<i>userChar</i> . Initialize (<i>chan</i>)
Variable	(Type) - Description
<i>userChar</i>	ECalUserCharacterizer (object)
<i>chan</i>	(Long) Channel number of a calibrated S-parameter channel.
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>userchar.Initialize(2)</code>
C++ Syntax	HRESULT put_Initialize(long chanNum);
Interface	IECalUserCharacterizer

Last Modified:

4-Nov-2008 New topic (8.33)

IsLibraryImported Method

Description Returns whether a DLL has been imported into the PNA.

VB Syntax *flag* = *equation*.**IsLibraryImported** *location*

Variable [\(Type\)](#) - Description

flag **(Boolean)**

True - DLL has been imported.

False - DLL has NOT been imported.

equation A [MeasurementEquation](#) object

location **(string)** – Full path and filename of the *.dll.

Return Type Variant boolean

Default Not Applicable

Examples `flag=equation.IsLibraryImported "C:/Program Files/Agilent/Network Analyzer/UserFunctions/Expansion.dll"`

C++ Syntax HRESULT IsLibraryImported(BSTR filename, VARIANT_BOOL* pImported);

Interface IMeasurementEquation2

Last Modified:

10-Jan-2011 New topic

Item Method

Description Returns an object from the collection of objects.

Notes

- The order of objects within a collection cannot be assumed.
- Most, but not all, PNA Collections are '1-based'

VB Syntax *Object*[**Item**](*n*)

Variable (**Type**) - Description

Object Any of the following (**collections**):

- [CalFactorSegments collection](#)
- [CalFactorSegmentsPMAR Collection](#)
- [Cal Sets collection](#)
- [Channels collection](#)
- [E5091Testset collection](#)
- [ECalModules Collection](#)
- [ExternalDevices Collection](#)
- [ExternalTestsets collection](#)
- [FOM collection](#)
- [GuidedCalibrationPowerSensors Collection](#)
- [LimitTest collection](#)
- [Measurements collection](#)
- [NaWindows collection](#)
- [PowerLossSegments collection](#)
- [PowerLossSegmentsPMAR_Collection](#)
- [PowerSensors collection](#)
- [Segments collection](#)

- [Traces collection](#)
- [PowerMeterInterfaces Collection](#)

[Learn more about collections in the PNA](#)

.Item Optional - Item is the default property of a collections object and therefore can be called implicitly. For example, the following two commands are equivalent:

```
Channels.Item(3).Averaging = 1
Channels(3).Averaging = 1
```

n (variant) - number of the item in the collection.

- The **Measurements, Traces, and FOM** collections allow you to specify the name of the measurement as a string. For example:
`measCollection("CH_S11_1").InterpolateMarkers`
- The [Cal Sets collection](#) allows you to specify the name of the cal set. For example:
`Calsets("MyCalSet").Description = "New Description"`
- The [ExternalDevices Collection](#) allows you to specify the name of the external device. For example:
`Set extDev = externalDevices.Item("NewPMAR")`
- The [GuidedCalibrationPowerSensors Collection](#) allows you to specify the name of the Power Sensor. For example:
`Set PowerSensor = GuidedCalibrationPowerSensors.Item.Name = "26GHzPowerSensor"`

Return Type (Object)

Default Not Applicable

Examples

```
For i = 1 to Traces.Count 1
  Traces.Item(i).YScale = .5dB
Next i
```

C++ Syntax HRESULT Item(VARIANT index, <interface>** pItem)

Interfaces All listed above.

Last Modified:

8-Feb-2011 Added GuidedCalibrationPowerSensors
 10-Sep-2008 Added CalSets name note

LANConfigurationInitialize Method

Description	Performs an initialization (reset) of the PNA's LAN configuration, as dictated by Section 8.14 of the LAN eXtensions for Instrumentation (LXI) standard (Version 1.1). This performs the same operation as pressing the LAN Reset button on the PNA LAN Status dialog .
VB Syntax	<i>app</i> .LANConfigurationInitialize
Variable	(Type) - Description
	<i>app</i> An Application (object)
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>app.LANConfigurationInitialize</code>
C++ Syntax	HRESULT LANConfigurationInitialize ();
Interface	IApplication13

Last Modified:

2-Jun-2008 MX New topic

LaunchCalWizard Method

Description Launches the S-parameter Cal Wizard on the PNA and does not return until the Cal Wizard is dismissed.

To launch the Cal Wizard for a PNA Application, use the [LaunchDialog Method](#).

Note: The Cal Wizard operates on the active measurement. Therefore, activate the measurement to be calibrated before launching the Cal Wizard.

VB Syntax `success = app.LaunchCalWizard (newCS)`

Variable [\(Type\)](#) - Description

`success` **(boolean)** - variable to store the returned value

True - The Cal was completed

False - The Cal was canceled without completing the calibration.

`app` An [Application](#) **(object)**

`newCS` **(boolean)**

True - Cal will be performed on a new Cal Set.

False - Cal will be performed using the existing Cal Set assigned to the channel. If no Cal Set is found, a new Cal Set will be created.

Return Type Boolean

Default Not Applicable

Example

```
dim bSuccess as boolean
dim bNewCalset as boolean
bNewCalSet = false
bSuccess = app.LaunchCalWizard( bNewCalSet)
```

C++ Syntax HRESULT LaunchCalWizard(VARIANT_BOOL bCalsuccess)

Interface IApplication

Last Modified:

5-Sep-2008 S-parameter only

LaunchDialog Method

Description	<p>Launches the specified dialog box.</p> <p>The Calibration Wizard dialog that appears depends on the active channel. For example, if a Gain Compression channel is active, then the GCA Cal Wizard appears. Use meas.Activate to activate a measurement and channel.</p> <p>Remote operation returns after the dialog is dismissed.</p> <p>To invoke the Cal Wizard and have it return immediately, then use Syst:Corr:Wiz with the SCPI Parser object.</p>
VB Syntax	<code>app.LaunchDialog dialog, [data]</code>
Variable	(Type) - Description
<i>app</i>	An Application (object)
<i>dialog</i>	(String) Dialog box to launch. Choose from:
	"SourcePowerCal" See this dialog.
	"PowerMeterSettings" See this dialog.
	"PathConfiguration" See this dialog.
	"CalibrationWizard" Depends on the channel
	"CalibrationSelection" See this dialog.
<i>[data]</i>	(Optional argument) Reserved for future use.
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>app.LaunchDialog "SourcePowerCal"</code>
C++ Syntax	LaunchDialog(BSTR dialog, [defaultvalue(0)] VARIANT dialogData)
Interface	IApplication11

[See the PNA Object Model](#)

4-Sep-2008 Added new dialogs
1-Jun-2007 Added optional argument
16-Feb-2007 MX New topic

LaunchPowerMeterSettingsDialog Method

Description	Launches the Power Meter Settings dialog on the PNA. Changing certain values from that dialog will change values of the corresponding properties on this COM object.
VB Syntax	<i>pwrCal</i> . LaunchPowerMeterSettingsDialog
Variable	(Type) - Description
<i>pwrCal</i>	A SourcePowerCalibrator (object)
Return Type	None
Default	Not Applicable
Examples	<code>powerCalibrator.LaunchPowerMeterSettingsDialog</code>
C++ Syntax	HRESULT put_LaunchPowerMeterSettingsDialog();
Interface	ISourcePowerCalibrator2

LoadConfiguration Method

Description Loads the named configuration onto the specified channel.

Note: Loading a stored configuration will over-write MANY RF and [IF path configuration](#) settings. Make your measurement settings AFTER recalling a stored configuration, NOT before.

Use [Configurations Method](#) to return the configuration names that are stored on the PNA.

VB Syntax `pathMgr.LoadConfiguration ch, name`

Variable [\(Type\) - Description](#)

`pathMgr` [PathConfigurationManager](#) (object)

`ch` (**Long**) Channel number of the configuration to be saved.

`name` (**String**) Configuration name. "Default" is the default factory configuration.

Return Type Not Applicable

Default Not Applicable

Examples `path.LoadConfiguration 2, "myMixer"`

C++ Syntax `HRESULT LoadConfiguration (long channelNum, BSTR configName);`

Interface `IPathConfigurationManager`

Last Modified:

18-Oct-2012 Added note

14-Dec-2006 MX New topic

LoadENRFile Method

Description	Loads an ENR file from disk into PNA memory. This file is typically provided by the manufacturer of the noise source.
VB Syntax	<i>enr</i> .LoadENRFile (<i>filename</i>)
Variable	(Type) - Description
<i>enr</i>	An ENRFile (object)
<i>filename</i>	(String) - Absolute path and filename of the ENR file.
Return Type	Not Applicable
Default	Not Applicable
Examples	See example program
C++ Syntax	HRESULT LoadENRFile(BSTR filename);
Interface	IENRFile

Last Modified:

2-Aug-2007 MX New topic

LoadFile Method

Description	Recalls an external device configuration file from the PNA hard drive. Currently, only DC Supply and DC Meter configuration files are supported. See more DC Device commands .
VB Syntax	<i>extDev</i> . LoadFile (<i>filename</i>)
Variable	(Type) - Description
<i>extDev</i>	An ExternalDevice Object
<i>filename</i>	(String) File name and .xml extension of the external device configuration file. Currently, only DC Supply and DC Meter configuration files are supported. See more DC Device commands .
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>extDev.LoadFile "MyDCSupply"</code>
C++ Syntax	HRESULT LoadFile(BSTR newVal)
Interface	IExternalDevice

Last Modified:

11-Apr-2012 New topic

Write only

LoadFile Method

Description Loads a previously-configured mixer attributes file (.mxr)

VB Syntax *obj*.LoadFile (*filename*)

Variable [\(Type\)](#) - Description

obj A [Mixer Interface](#) pointer to the [Measurement](#) (object)

Or

A [Converter Object](#)

filename (String) Full path, file name, and .mxr extension of the mixer attributes file.

Files are typically stored in "C:/Program Files/Agilent/Network Analyzer/Documents".

Return Type String

Default Not Applicable

Examples

```
mixer.LoadFile ("c:\users\public\network  
analyzer\documents\myMixer.mxr")
```

C++ Syntax HRESULT LoadFile(BSTR newVal)

Interface IMixer

IConverter

Last Modified:

2-Feb-2009 Added converter

LoadTheme Method

Description	Load a color theme from a disc file.
VB Syntax	<i>colors</i> . LoadTheme (<i>filename</i>)
Variable	(Type) - Description
<i>colors</i>	A ComColors (object)
<i>filename</i>	(String) - Path and filename of the theme to load.
Return Type	Not Applicable
Default	Not Applicable
Examples	<pre>colors.LoadTheme = ("c:/Program Files/Agilent/Network Analyzer/Colors/Theme1.colors")</pre>
C++ Syntax	HRESULT LoadTheme(BSTR filename);
Interface	IColors

Last Modified:

7-Aug-2009 MX New topic

ManualTrigger Method

Description Triggers the analyzer when [TriggerSetup.Source](#) = naTriggerManual.

Note: An **SMC Fixed Output** measurement cannot be triggered using this command. For more information, see the [example program](#).

VB Syntax `app.ManualTrigger [sync],[timeout]`

Variable [\(Type\) - Description](#)

app An [Application](#) (object)

[sync] (boolean) - Optional argument.

A variable set to either True or False.

True - The analyzer waits until the trigger is completed to process subsequent commands.

False - Subsequent commands are processed immediately (the default setting).

timeout (long) - Optional argument.

If *sync* is true, *timeout* sets the amount of time the PNA will wait until continuing program execution. Units are milliseconds. A value of -1 (the default setting) causes the PNA to wait indefinitely.

If *sync* is False, the timeout setting is ignored.

Return Type Not Applicable

Default Not Applicable

Examples

```
' After Manual trigger is executed, the PNA will wait 1 second
to continue program execution
Dim wait as Boolean
wait = True
app.ManualTrigger wait, 1000
```

C++ Syntax HRESULT ManualTrigger(VARIANT_BOOL bSynchronize, long timeout)

Interface IApplication

Last Modified:

12-Jul-2007 [Modified link](#)

Move Method

Description	Moves a trace from one window to another.
VB Syntax	<i>trace</i> . Move (<i>winTo</i>)
Variable	(Type) - Description
<i>trace</i>	Trace (object)
<i>winTo</i>	(Long integer) Window number to move the trace to. If the window number does not exist, it will be created.
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>trace.Move 2 'Moves the trace to window 2</code>
C++ Syntax	HRESULT Move()
Interface	ITrace3

Last Modified:

25-Jul-2011 New topic

NetworkPortMap Method

Description Set the port mapping for a 4-port SNP file to be embedded.

To read port mapping, use [NetworkPortMapA](#), [NetworkPortMapB](#), [NetworkPortMapC](#), [NetworkPortMapD](#).

VB Syntax *fixture*.**NetworkPortMap** *network, inA, inB, outA, outB*

Variable **(Type) - Description**

fixture A [Fixturing](#) (object)

network Network position. Choose from **1** or **2**.

inA, inB, outA, outB Port Mapping. Use four port numbers in any order.

Return Type Not Applicable.

Default 1,2,3,4

Examples `fixture.NetworkPortMap 1,1,3,2,4`

C++ Syntax HRESULT NetworkPortMap(short Net, *Long inA, *Long inB,* Long outA,* Long outB,)

Interface IFixturing6

Last Modified:

10-Jan-2012 Major edits

16-Nov-2010 MX New topic

NextIFBandwidth Method

Description	A function that returns the Next higher IF Bandwidth value. Use to retrieve the list of available IFBandwidth settings.
VB Syntax	<i>chan</i> .Next_IFBandwidth <i>bw</i>
Variable	(Type) - Description
<i>chan</i>	A Channel (object)
<i>bw</i>	(double) - The argument that you use to send an IFBandwidth. The function uses this argument to return the Next higher IFbandwidth.
Return Type	Double
Default	Not Applicable
Examples	<pre>Public pnbw As Double 'declare variable outside of procedure pnbw = chan.IFBandwidth 'put the current IFBW in pnbw chan.Next_IFBandwidth pnbw 'function returns the Next higher IFBandwidth. chan.IFBandwidth = pnbw 'set IFBW to the Next value</pre>
C++ Syntax	HRESULT Next_IFBandwidth (double *pVal)
Interface	IChannel

NumberOfGroups Method

Description	Sets the number of trigger signals the channel will receive. After the channel has received that number of trigger signals, the channel switches to Hold mode.
VB Syntax	<code>chan.NumberOfGroups num, sync</code>
Variable	(Type) - Description
<i>chan</i>	A Channel (object)
<i>num</i>	(long integer) Number of trigger signals the channel will receive. Choose any number between 1 and 2 million.
<i>sync</i>	(boolean)
	Variable set to either:
	True - subsequent commands are not processed until the groups are complete. Do not use with manual trigger.
	False - subsequent commands are processed immediately.
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>chan.NumberOfGroups 5,False</code>
C++ Syntax	HRESULT NumberOfGroups(long count, VARIANT_BOOL bWait)
Interface	IChannel

OpenCalSet Method **Superseded**

Description This command is no longer necessary. The CalSet.get... and put... commands that required this command have been replaced,

Open the Cal Set to read/write a particular **CalType**. Learn more about [reading and writing Cal Data using COM](#).

This method is a prerequisite to several other Cal Set methods.

A Cal Set can contain more than one CalType. This method opens the Cal Set and allows access to a particular set of terms. Subsequent commands like [getErrorTerm](#) use this information to access the correct error terms in the Cal Set. For example:

```
cset.OpenCalSet (naCalType_TwoPortSOLT,3,2)
cset.PutErrorTerm(naDirectivity, 1, 1, Buffer)
```

The directivity error term for port 1 could belong to any number of caltypes: Full1Port (S11), Full2Port (12), Full2Port (13) or Full3Port (123). The **CalType and port** specifiers in OpenCalSet directs the uploaded directivity term to the correct set of error terms.

To close the Cal Set, see [CloseCalSet](#).

VB Syntax *CalSet.OpenCalSet (CalType, p1, p2, p3)*

Variable [\(Type\)](#) - **Description**

CalSet **(object)** - A Cal Set object

CalType (enum as naCalType) - type of correction to be applied. Choose from:

Caltype	p arguments required
0 - naCalType_Response_Open	p1
1 - naCalType_Response_Short	p1
2 - *naCalType_Response_Thru	p1 (rcv), p2 (src)
3 - *naCalType_Response_Thru_And_Isol	p1 (rcv), p2 (src)
4 - naCalType_OnePort	p1
5 - naCalType_TwoPort_SOLT	p1, p2
6 - naCalType_TwoPort_TRL	p1, p2
7 - naCalType_None	N/A
8 - naCalType_ThreePort_SOLT	p1, p2, p3
9 - Custom	N/A

10 - naCalType_FourPort_SOLT p1, p2, p3
(port 4 is assumed)

* order of port arguments is significant for these CalTypes

p1 **(long)** - required. This argument must be specified.

This specifies either:

- the one significant port for an open/short response cal or a 1 port cal.
- or one of the ports involved in a 2 or 3 port cal
- or the *receive* port for a thru response / thru-isolation cal.

p2 **(long)** - required for any caltype involving more than one port

This specifies either:

- one of the ports involved in a 2 or 3 port cal (order independent)
- or the *source* port for a thru response / thru-isolation cal

p3 **(long)** - required only for 3 port cal

This specifies either:

- one of the ports involved in a 3 port cal (order independent)

Return Type None

Default Not Applicable

Examples `CalSet.OpenCalSet naCalType_ThreePort_SOLT, 3,2,1`

C++ Syntax HRESULT OpenCalSet (naCalType, port1, [optional] port2, [optional] port3);

Interface ICalSet

Parse Method

Description Allows the use of COM to send a SCPI command. [See a C++ example](#) of how to return error information when using this command.

Note: The SCPIStringParser Methods can NOT be used with [SCPI Status Reporting](#). However, the *OPC? will work.

VB Syntax `scpi.Parse ("SCPI command")`

Variable [\(Type\)](#) - Description

`scpi` A [ScpiStringParser](#) (**object**)

`SCPI command` (**string**) - Any valid SCPI command

Return Type String

Default Not Applicable

Examples

```
Dim scpi As ScpiStringParser
Set scpi = app.ScpiStringParser
Dim startfreq As Double
startfreq = 100e6
'
scpi.Parse "Sens:Freq:Start " & startfreq 'Write

Dim str As String
str = scpi.Parse ("Sens:Freq:Start?") 'Read
```

C++ Syntax HRESULT Parse(BSTR SCPI_Command, BSTR *pQueryResponse)

Interface IScpiStringParser

Last Modified:

April 27, 2009 Added note
1-Jan-2007 Corrected example

Preset Method

Description **Application Object:** Deletes all traces and windows. In addition, resets the analyzer to factory defined default settings and creates an S11 measurement named "CH1_S11_1" in window 1.

Channel Object: Resets the channel (object) to factory defined default settings. Does NOT delete the current measurements or add a new measurement.

VB Syntax *app.Preset*
chan.Preset

Variable [\(Type\)](#) - Description

app An [Application](#) (**object**)

chan A Channel (**object**)

Return Type Not Applicable

Default Not Applicable

Examples `app.Preset`

C++ Syntax HRESULT Preset()

Interface IApplication
IChannel

PreviousIFBandwidth Method

Description	A function that returns the previous IF Bandwidth value. Use to retrieve the list of available IFBandwidth settings.
VB Syntax	<i>chan</i> .Previous_IFBandwidth <i>bw</i>
Variable	(Type) - Description
<i>chan</i>	A Channel (object)
<i>bw</i>	(double) - The argument that you use to send an IFBandwidth. The function uses this argument to return the previous IFbandwidth.
Return Type	Double
Default	Not Applicable
Examples	<pre>Public pnbw As Double 'declare variable outside of procedure PreBW = chan.IFBandwidth 'put the current IFBW in PreBW chan.Previous_IFBandwidth PreBW 'function returns the Previous IFBandwidth of the current one. chan.IFBandwidth = PreBW 'set IFBW to the previous value</pre>
C++ Syntax	HRESULT Previous_IFBandwidth (double *pVal)
Interface	IChannel

PrintToFile Method

Description Saves the screen image to a bitmap file.

VB Syntax *app.PrintToFile filename*

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

filename (**string**) Full path, file name, and extension of the screen image file.

Files are typically stored in "c:\users\public\network analyzer\documents".

Use one of the following extensions:

- .bmp - not recommended due to large file size
- .jpg - not recommended due to poor quality
- **.png - recommended**

Return Type Not Applicable

Default Not Applicable

Examples `app.PrintToFile "C:/Program Files/Agilent/Network Analyzer/Documents/myfile.png"`

C++ Syntax HRESULT PrintToFile(BSTR bstrFile)

Interface IApplication

PutComplex Method

Description Puts real and imaginary data into the specified location. This method forces the channel into Hold mode to prevent the input data from being overwritten. Learn more about [reading and writing Cal Data using COM](#).

Data put in the raw data store will be **re-processed** whenever a change is made to the measurement attributes such as format or correction.

Data put in the measurement results store will be **overwritten** by any measurement attribute changes.

See also [putNAComplex](#)

VB Syntax *measData.putComplex location, numPts, real(), imag(), [format]*

Variable [\(Type\) - Description](#)

measData An IArrayTransfer interface which supports the Measurement object

location (**enum NADataStore**) Where the Data will be put. Choose from:

0 - naRawData

1 - naCorrectedData

2 - naMeasResult

3 - naRawMemory

4 - naMemoryResult

5 - naDivisor - When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using [DataToDivisor Method](#).

numPts (**long integer**) - Number of data points in the channel

real() (**single**) - Array containing real data values

imag() (**single**) - Array containing imaginary data values

format (**enum NADataFormat**) optional argument - display format of the real and imaginary data. Only used if destination is naMeasResult or naMemoryResult buffer. If unspecified, data is assumed to be in naDataFormat_Polar

0 - naDataFormat_LinMag

1 - naDataFormat_LogMag

- 2 - naDataFormat_Phase
- 3 - naDataFormat_Polar
- 4 - naDataFormat_Smith
- 5 - naDataFormat_Delay
- 6 - naDataFormat_Real
- 7 - naDataFormat_Imaginary
- 8 - naDataFormat_SWR
- 9 - naDataFormat_PhaseUnwrapped
- 10 - naDataFormat_InverseSmith
- 11 - naDataFormat_Kelvin
- 12 - naDataFormat_Fahrenheit
- 13 - naDataFormat_Centigrade

[Learn more about Data Format.](#)

Return Type Not Applicable

Default Not Applicable

Examples

```
Dim measData As IArrayTransfer
Set measData = app.ActiveMeasurement

measData.putComplex naMemoryResult, 201,
real(0), imag(0), naDataFormat_SWR
```

C++ Syntax HRESULT putComplex(tagNADataStore DataStore, long lNumValues, float* pReal, float* pImag, tagDataFormat displayFormat)

Interface IArrayTransfer

Last Modified:

1-Oct-2007 Added temperature formats

PutDataComplex Method

Description	Puts complex data into the specified location. This method forces the channel into Hold mode to prevent the input data from being overwritten.
VB Syntax	<i>meas.putDataComplex location, data</i>
Variable	(Type) - Description
<i>meas</i>	A measurement (object)
<i>location</i>	(enum NADataStore) Where the Data will be put. Choose from: 0 - naRawData 1 - naCorrectedData 2 - naMeasResult - Valid ONLY when the display format is either Polar or Smith Chart. 3 - naRawMemory - See note below. 4 - naMemoryResult 5 - naDivisor - When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using DataToDivisor Method .
	Note: When putting data into 3 - naRawMemory : <ol style="list-style-type: none">1. Put the analyzer in hold mode2. Call DataToMemory to initialize a memory buffer3. Call putDataComplex(naRawMemory, data) This ensures that the memory buffer is appropriately initialized before receiving new data.
<i>data</i>	(variant) - A two-dimensional variant array. Note: All buffers except naMeasResult and naMemoryResult require Complex data
Return Type	Not Applicable

Default Not Applicable

Examples

```
' Put 201 points worth of raw (complex) data into the
measurement
' Note that an array of complex numbers is represented by a 2-D
array where the first rank is the number of points, and the 2nd
rank is always size 2 (max index 1) representing the Real and
Imag parts of the complex number.

' complex array of data (2nd dimension of size 2 represents
Re/Im
Dim data(200,1) )
For i = 0 to 200
' Set Real part of data point i
data(i,0) = i/200;
' Set Imag part of data point i
data(i,1) = i/200;
Next
app.ActiveMeasurement.putDataComplex naRawData, data
```

C++ Syntax HRESULT putDataComplex(tagNADataStore DataStore, VARIANT
complexData)

Interface IMeasurement

Last Modified:

7-Aug-2012 Added naMeasResult note (DS)

PutENRData Method

Description	Write ENR calibration data to PNA memory. All of the frequency and ENR data must be sent at the same time.
VB Syntax	<i>enr.PutENRData (vData)</i>
Variable	(Type) - Description
<i>enr</i>	An ENRFile (object)
<i>vData</i>	(Variant array) -ENR data. Frequency value in Hz, followed by corresponding ENR value in dB. Enter as many data pairs as necessary.
Return Type	Not Applicable
Default	Not Applicable
Examples	See example program
C++ Syntax	HRESULT PutENRData(VARIANT vdata);
Interface	IENRFile

Last Modified:

2-Aug-2007 MX New topic

PutErrorTerm Method - Superseded

Description **Note:** This command is replaced by [PutErrorTermByString](#)

Puts variant error term data into the error-correction buffer.

[Learn about reading and writing Calibration data.](#)

VB Syntax *cal*.**putErrorTerm**(*term*,*rcv*, *src*, *data*)

Variable [\(Type\)](#) - Description

cal A Calibrator (**object**)

term (**enum As NaErrorTerm**)

naErrorTerm_Directivity_Isolation

naErrorTerm_Match

naErrorTerm_Tracking

rcv (**long integer**) - Receiver Port

src (**long integer**) - Source Port

data (**variant**) Error term data in a two-dimensional array (0:1, 0:numpts-1).

To get this	Specify these parameters:		
Error Term	<i>term</i>	<i>rcv</i>	<i>src</i>
Fwd Directivity	naET_Directivity Isolation	1	1
Rev Directivity	naET_Directivity Isolation	2	2
Fwd Isolation	naET_Directivity Isolation	2	1
Rev Isolation	naET_Directivity Isolation	1	2
Fwd Source Match	naErrorTerm_Match	1	1
Rev Source Match	naErrorTerm_Match	2	2
Fwd Load Match	naErrorTerm_Match	2	1
Rev Load Match	naErrorTerm_Match	1	2
Fwd Reflection Tracking	naErrorTerm_Tracking	1	1
Rev Reflection Tracking	naErrorTerm_Tracking	2	2
Fwd Trans Tracking	naErrorTerm_Tracking	2	1
Rev Trans Tracking	naErrorTerm_Tracking	1	2
Fwd Trans Tracking	naErrorTerm_Tracking	2	1

Return Type Not Applicable

Default Not Applicable

Examples `Dim varError As Variant
varError = cal.putErrorTerm (naErrorTerm_Tracking,2,1,VarData)`

C++ Syntax HRESULT putErrorTerm(tagNAErrorTerm ETerm, long ReceivePort, long SourcePort, VARIANT varData)

Interface ICalibrator

PutErrorTerm Method **Superseded**

Description	<p>This command is replaced with PutErrorTermByString</p> <p>Puts error term data into the Cal Set.</p> <p>Learn more about Reading and Writing Cal Data</p> <p>See examples of Reading and Writing Cal Set Data</p>
VB Syntax	<code>CalSet.putErrorTerm (term, rcv, src, data)</code>
Variable	(Type) - Description
<i>CalSet</i>	(Object) A CalSet Object
<i>term</i>	(enum As NAErrorTerm2) Error Term. Choose from: 0 - naET_Directivity (src = rcv) 1 - naET_SourceMatch (src = rcv) 2 - naET_ReflectionTracking (src = rcv) 3 - naET_TransmissionTracking (src ≠ rcv) 4 - naET_LoadMatch (src ≠ rcv) 5 - naET_Isolation (src ≠ rcv)
<i>rcv</i>	(long integer) - Receiver Port
<i>src</i>	(long integer) - Source Port
<i>data</i>	(variant) Error term data in a two-dimensional array (0:1, 0:numpts-1). The data must be complex pairs.
	Note: See also PutErrorTermComplex on the ICalData2 interface to avoid using the variant data type.
Return Type	Not Applicable
Default	Not Applicable
Examples	See an Example
C++ Syntax	HRESULT putErrorTerm(tagNAErrorTerm2 ETerm, long ReceivePort, long SourcePort, VARIANT varData)

Interface ICalSet

PutErrorTermByString Method

Description	Puts error term data into the Cal Set. Learn more about Reading and Writing Cal Data See examples of Reading and Writing Cal Set Data
VB Syntax	<i>calSet</i> . PutErrorTermByString (<i>errorName</i> , <i>vdata</i>)
Variable	(Type) - Description
<i>calSet</i>	(Object) A CalSet Object
<i>errorName</i>	(String) The string name used to identify a particular error term in the Cal Set. An example string for port 3 directivity in a full 2 port cal might be "Directivity(3,3)". To determine the string names of error terms, see GetErrorTermList2 .
<i>vdata</i>	(Variant) This data array is usually two dimensional. Each element is a type single. The two elements represent the real and imaginary parts of a complex pair. Note: This structure is compatible with scripting clients who can only use variants. For alternative methods that use typed arrays, see ICalData3.
Return Type	Not Applicable
Default	Not Applicable
Examples	See an Example
C++ Syntax	HRESULT PutStandardByString(BSTR bufferName, VARIANT vardata)
Interface	ICalSet2

PutErrorTermComplex Method **Superseded**

Description	Note: This command is replaced by PutErrorTermComplexByString Puts error term data into the error-correction data buffer. Learn more about reading and writing Cal data using COM
VB Syntax	<code>data.putErrorTermComplex term, rcv, src, numPts, real(), imag()</code>
Variable	(Type) - Description
<i>data</i>	An ICalData pointer to the Calibrator object
<i>term</i>	(enum NAErrorTerm) - The error term to be retrieved. Choose from: <ul style="list-style-type: none">• naErrorTerm_Directivity_Isolation• naErrorTerm_Match• naErrorTerm_Tracking
<i>rcv</i>	(long integer) - Receiver Port
<i>src</i>	(long integer) - Source Port
<i>numPts</i>	(long integer) - number of data points in the array
<i>real()</i>	(single) - array containing the real part of the calibration data. One-dimensional: the number of data points.
<i>imag()</i>	(single) - array containing the imaginary part of the calibration data. One-dimensional: the number of data points.

	To get this	Specify these parameters:		
	Error Term	<i>term</i>	<i>rcv</i>	<i>src</i>
	Fwd Directivity	naET_Directivity Isolation	1	1
	Rev Directivity	naET_Directivity Isolation	2	2
	Fwd Isolation	naET_Directivity Isolation	2	1
	Rev Isolation	naET_Directivity Isolation	1	2
	Fwd Source Match	naErrorTerm_Match	1	1
	Rev Source Match	naErrorTerm_Match	2	2
	Fwd Load Match	naErrorTerm_Match	2	1
	Rev Load Match	naErrorTerm_Match	1	2
	Fwd Reflection Tracking	naErrorTerm_Tracking	1	1
	Rev Reflection Tracking	naErrorTerm_Tracking	2	2
	Fwd Trans Tracking	naErrorTerm_Tracking	2	1
	Rev Trans Tracking	naErrorTerm_Tracking	1	2
	Fwd Trans Tracking	naErrorTerm_Tracking	2	1
Return Type	Not Applicable			
Default	Not Applicable			
Examples	<pre>Dim eData As ICalData Set eData = chan.Calibrator eData.putErrorTermComplex naErrorTerm_Directivity_Isolation, 1, 1, 201, rel(0), img(0)</pre>			
C++ Syntax	HRESULT putErrorTermComplex(tagNAErrorTerm ETerm, long ReceivePort, long SourcePort, long* pNumValues, float* pReal, float* pImag)			
Interface	ICalData			

PutErrorTermComplex Method **Superseded**

Description	This command is replaced with PutErrorTermComplexByString Puts error term data into the Cal Set. Learn more about Reading and Writing Cal Data See examples of Reading and Writing Cal Set Data
VB Syntax	<code>data.putErrorTermComplex term, rcv, src, numPts, real(), imag()</code>
Variable	(Type) - Description
<i>data</i>	An ICalData2 pointer to a Cal Set object
<i>term</i>	(enum NAErrorTerm2) - The error term to be written. Choose from: 0 - naET_Directivity 1 - naET_SourceMatch 2 - naET_ReflectionTracking 3 - naET_TransmissionTracking 4 - naET_LoadMatch 5 - naET_Isolation
<i>rcv</i>	(long) - Receiver Port
<i>src</i>	(long) - Source Port
<i>numPts</i>	(long) - number of data points in the real and imaginary arrays.
<i>real()</i>	(single) - array containing the real part of the calibration data. One-dimensional: the number of data points.
<i>imag()</i>	(single) - array containing the imaginary part of the calibration data. One-dimensional: the number of data points.
Return Type	Not Applicable
Default	Not Applicable
Examples	<pre>Dim eData As ICalData2 Set eData = app.GetCalManager.Cal Sets.Item(1) eData.putErrorTermComplex naET_LoadMatch, 1, 2, numpts, rel(0), img(0)</pre>

C++ Syntax HRESULT putErrorTermComplex(tagNAErrorTerm2 ETerm, long ReceivePort, long SourcePort, long* pNumValues, float* pReal, float* pImag)

Interface ICalData2

PutErrorTermComplexByString Method

Description Puts error term data into the Cal Set.

Learn more about [Reading and Writing Cal Data](#)

See examples of [Reading](#) and [Writing](#) Cal Set Data

VB Syntax `ICalData3.PutErrorTermComplexByString errorName, lnumPoints, real(0), imag(0)`

Variable [\(Type\) - Description](#)

ICalData3 An [ICalData3](#) pointer to a Cal Set object.

errorName **(String)** The string name used to identify a particular error term in the Cal Set. An example string for port 3 directivity in a full 2 port cal might be "Directivity(3,3)". To determine the string names of error terms, see [GetErrorTermList2](#).

lnumPoints **(Long)** The number of data points in the real and imaginary arrays.

real **(Single)** The real component of the complex data.

imag **(Single)** The imaginary component of the complex data.

Note: The size of the real and imaginary arrays should be the same.

Return Value Not Applicable

Default Not Applicable

Examples [See example](#)

C++ Syntax `HRESULT PutErrorTermComplexByString(BSTR bufferName, long lnumPoints, float* real, float* imag);`

Interface ICalData3

PutErrorTermStimulus Method

Description Adds stimulus data to the specified buffer. The size of vdata must agree with the size of the complex data already attached to the buffer or an error will be generated.

See Also: [GetErrorTermStimulus Method](#)

Learn more about [Reading and Writing Cal Data](#)

See examples of [Reading](#) and [Writing](#) Cal Set Data

VB Syntax *calSet*.**PutErrorTermStimulus**(*bufferName*, *vdata*)

Variable [\(Type\) - Description](#)

calSet **(Object)** A [CalSet](#) Object

bufferName **(String)** The string name used to identify a particular error term in the Cal Set. An example string for port 3 directivity in a full 2 port cal might be "Directivity(3,3)". To determine the string names of error terms, see [GetErrorTermList2](#).

vdata **(Variant)** Safearray of variants (doubles).

Return Type Not Applicable

Default Not Applicable

Examples The sequence is:

```
complexData = calset1.GetErrorTermByString(0, BufferName)
frequencyData = calset1.GetErrorTermStimulus(0, BufferName)
// manipulate complex data here
Calset2.PutErrorTermByString(BufferName, manipulatedComplexData)
Calset2.PutErrorTermStimulus(BufferName, frequencyData);
```

C++ Syntax HRESULT PutErrorTermStimulus (BSTR bufferName, VARIANT vardata)

Interface ICalSet7

Last modified:

2-May-2011 New topic

PutScalar Method

Description Puts Scalar data in the Measurement Result buffer. The putScalar array is not processed by the analyzer; it is just displayed. Any change to the measurement state (changing the format, for example) will cause the putScalar data to be overwritten with the data processed from the raw data buffer.

VB Syntax *measData.putScalar, format, numPts, data*

Variable [\(Type\) - Description](#)

measData An IArrayTransfer interface which supports the Measurement object.

format **(enum NADataFormat)** Format of the data. Choose from:

- 0 - naDataFormat_LinMag
- 1 - naDataFormat_LogMag
- 2 - naDataFormat_Phase
- 3 - naDataFormat_Polar
- 4 - naDataFormat_Smith
- 5 - naDataFormat_Delay
- 6 - naDataFormat_Real
- 7 - naDataFormat_Imaginary
- 8 - naDataFormat_SWR
- 9 - naDataFormat_PhaseUnwrapped
- 10 - naDataFormat_InverseSmith
- 11 - naDataFormat_Kelvin
- 12 - naDataFormat_Fahrenheit
- 13 - naDataFormat_Centigrade

[Learn more about Data Format.](#)

Note: Smith, InverseSmith, and Polar formats are not allowed.

numPts (**integer**) - Number of values. Usually the number of points in the trace (chan.NumberOfPoints).

data (**single**) - A one-dimensional array of Scalar data matching the number of points in the current measurement.

Return Type Not Applicable

Default Not Applicable

Examples

```
Dim measData As IArrayTransfer
Set measData = app.ActiveMeasurement

measData.putScalar naDataFormat_LogMag, 201, dScalar(0)
```

C++ Syntax HRESULT putScalar/(tagDataFormat eFormat, long lNumValues, float* pArrayOfScalar)

Interface IArrayTransfer

Last Modified:

1-Oct-2007 Added temperature formats

PutNAComplex Method

Description Puts complex data into the specified location. This method forces the channel into Hold mode to prevent the input data from being overwritten. The data is processed and displayed.

Data put in the naRawData store will be **re-processed** whenever a change is made to the measurement attributes such as format or correction.

Data put in the naMeasResult store will be **overwritten** by any measurement attribute changes (such as moving a marker).

Note: This method uses NAComplex which is a user-defined data type. If you cannot or prefer not to use this data type, use the [putComplex](#) method.

VB Syntax *measData*.putNAComplex *location*, *numPts*, *data*, [*format*]

Variable [\(Type\) - Description](#)

measData An IArrayTransfer interface which supports the Measurement object

location (**enum NADataStore**) Where the Data will be put. Choose from:

0 - naRawData

1 - naCorrectedData

2 - naMeasResult

3 - naRawMemory

4 - naMemoryResult

5 - naDivisor - When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using [DataToDivisor Method](#).

numPts (**long integer**) - Number of data points in the channel

data (**NAComplex**) - A one-dimensional array of Complex data matching the number of points in the current measurement.

format (**enum NADataFormat**) - Optional argument. Format of the data. If unspecified, naDataFormat_Polar is assumed. Only used when the destination store is naMeasResult or naMemoryResult.

0 - naDataFormat_LinMag

- 1 - naDataFormat_LogMag
- 2 - naDataFormat_Phase
- 3 - naDataFormat_Polar
- 4 - naDataFormat_Smith
- 5 - naDataFormat_Delay
- 6 - naDataFormat_Real
- 7 - naDataFormat_Imaginary
- 8 - naDataFormat_SWR
- 9 - naDataFormat_PhaseUnwrapped
- 10 - naDataFormat_InverseSmith
- 11 - naDataFormat_Kelvin
- 12 - naDataFormat_Fahrenheit
- 13 - naDataFormat_Centigrade

[Learn more about Data Format.](#)

Return Type Not Applicable

Default Not Applicable

Examples

```
Dim measData As IArrayTransfer
Set measData = app.ActiveMeasurement

measData.putNAComplex naMemoryResult, 201, dRawComplex(0)
```

C++ Syntax HRESULT putNAComplex(tagNADataStore DataStore, long INumValues, TsComplex* pArrayOfComplex, tagDataFormat displayFormat)

Interface IArrayTransfer

Last Modified:

1-Oct-2007 Added temperature formats

put_Output Method

Description	Writes a TTL HI or TTL Low to output pins 3 or 4 of the Material Handler IO connector. Each pin also has a latched output which is written to with USER. With the latched (USER) outputs, the value is not applied to the associated pin until a positive edge is detected at INPUT1 (pin 2).
VB Syntax	<i>handlerIo.put_Output (pin) = value</i>
Variable	(Type) - Description
<i>handlerIo</i>	(object) - A HandlerIO object
<i>pin</i>	(enum as NAMatHandlerOutput) - pin to write data to. Choose from: naOutput1 - (0) - pin3 naOutput1User (1) - pin3 latched (applied to pin 3 on positive edge of Input1-pin2) naOutput2 (2) - pin4 naOutput2User (3) - pin4 latched (applied to pin 4 on positive edge of Input1-pin2)
<i>value</i>	(Variant) Value to write to the selected pin. Choose from 0 - TTL LOW 1 - TTL HIGH
Return Type	Not Applicable
Default	0
Examples	<pre>handlerIo.put Output(naOutput1)= 1</pre>
C++ Syntax	HRESULT put_Output (tagNAMatHandlerOutput Output, VARIANT Data);
Interface	IHWMaterialHandlerIO

Write-only

put_OutputVoltage Method

Description **E836x and PNA-L:** Sets voltages on the DAC/Analog Output 1|2 of the Auxiliary IO connector.

PNA-X: Sets voltage on the [Power I/O connector](#) AnalogOut1|2.

Read output voltages using [get OutputVoltage Method](#).

VB Syntax *AuxIO.put_OutputVoltage output, voltage*

Variable [\(Type\) - Description](#)

AuxIO **(object)** - A Hardware Auxiliary Input / Output object

output **(variant)** Number of the output DAC to write voltage to. Choose from:

1 Output 1 (Aux I/O pin 3) and (Power I/O pin 3)

2 Output 2 (Aux I/O pin 2) and (Power I/O pin 4)

voltage **(double)** Voltage to write to the output DAC. Choose a voltage from -10 to 10

Return Type None

Default None

Examples `HWAuxIO.put_OutputVoltage 1,9 'set Analog Out1 to +9v`

C++ Syntax HRESULT put_OutputVoltage (VARIANT Output, double Voltage);

Interface IHWAuxIO

Last Modified:

10-Jul-2007 Added PNA-X capability

put_OutputVoltageMode Method

Description This command sets the mode of the selected "Analog Out" line on the [Power I/O connector](#). The modes give the user the option to have the requested voltage applied immediately or not until the sweep is done. To read the mode on each output use [get_OutputVoltageMode Method](#).

VB Syntax *auxIo.put_OutputVoltageMode (output, mode)*

Variable [\(Type\) - Description](#)

auxIo **(Object)** An AuxIO object

output Analog Output to receive mode setting. Choose from **1** or **2**

mode **(enum NAOutputVoltageMode)**

naWaitEOS - While in this mode any voltage changes sent to the selected analog out will only get applied to the output between sweeps.

naNoWait - While in this mode any voltage changes sent to the selected analog out will occur right away without waiting until the end of a sweep, the voltage gets applied immediately.

Return Type NAOutputVoltageMode

Default naWaitEOS

Examples `auxIo.put_OutputVoltageMode 1, naWaitEOS 'Write`

C++ Syntax HRESULT put_OutputVoltageMode(VARIANT Output, tagNAOutputVoltageMode dNewMode);

Interface IHWAuxIO

Last Modified:

10-Jul-2007 Added PNA-X capability

put_Port Method

Description Writes a value to the specified port. Use the [get_Port](#) Method to read the settings from the "readable" ports (C, D, E).

VB Syntax *handlerIo.put_Port (port, value)*

Variable [\(Type\) - Description](#)

handlerIo **(object)** - A HandlerIO object

port **(enum as NAMatHandlerPort)** - port to put data into. Choose from:

naPortA - (0)

naPortB - (1)

naPortC - (2)

naPortD - (3)

naPortE - (4)

naPortF - (5)

naPortG - (6)

naPortH - (7)

value The number of the data bits to set. The following table shows what the *value* represents:

Note: When writing to port G, port C must be set to output mode

When writing to port H, both port C and port D must be set to output mode. Use Port Mode Property

Port	Max allowable <num>	MSB.....LSB 23.....0	
A	255	A7...A0	Write-only
B	255	B7...B0	Write-only
C	15	C3...C0	Read-Write
D	15	D3...D0	Read-Write
E	255	D3...D0 + C3...C0	Read-Write
F	65535	B7...B0 + A7...A0	Write-only
G	1048575	C3...C0 + B7...B0 + A7...A0	Write-only
H	16777215	D3...D0 + C3...C0 + B7...B0 + A7...A0	Write-only

Return Type Not Applicable

Default Not Applicable

Examples `handlerIo.put_Port (naPortB, 1)`

C++ Syntax HRESULT put_Port (tagNAMatHandlerPort Port, VARIANT Data);

Interface IHWMaterialHandlerIO

Last Modified:

21-Mar-2008 Fixed syntax

Write-only

put_PortCData Method

Description Writes a 4-bit value to Port C on the Aux I/O connector (pins 22-25) and the Material Handler IO (pins 21-24 Anritsu) - (pins 22-25 Avantest).

Note: These lines are connected to both the Handler IO and Aux IO in the PNA. Therefore, this command will affect both of these connectors in the same way.

VB Syntax *AuxIO.put_PortCData num*

Variable [\(Type\) - Description](#)

AuxIO **(object)** - A Hardware Auxiliary Input / Output object

num **(variant)** - 4 bit binary value. Choose from 0-15

Return Type None

Default None

Examples `HWAuxIO.put_PortCData 15 'If Positive Logic, Port C lines C0, C1, C2, C3 go High. If Negative Logic, they go Low.`

C++ Syntax HRESULT put_PortCData(VARIANT Data);

Interface IHWAuxIO

PutDataScalar Method

Description Puts formatted variant scalar data into the measurement result buffer. The data will be immediately processed and displayed. Subsequent changes to the measurement state will be reflected on the display.

Always precede this command by setting the format on the measurement to be consistent with the format of the data being sent to the analyzer. In this way, the display annotation will be correct.

Execution of this command does not change the display format.

VB Syntax *meas.putDataScalar format, data*

Variable [\(Type\)](#) - Description

meas A measurement (**object**)

format (**enum NADataFormat**) Format of the data. **This value is presently ignored by the PNA.** Data is always presented in the current format.

Choose from:

0 - naDataFormat_LinMag

1 - naDataFormat_LogMag

2 - naDataFormat_Phase

3 - naDataFormat_Polar

4 - naDataFormat_Smith

5 - naDataFormat_Delay

6 - naDataFormat_Real

7 - naDataFormat_Imaginary

8 - naDataFormat_SWR

9 - naDataFormat_PhaseUnwrapped

10 - naDataFormat_InverseSmith

11 - naDataFormat_Kelvin

12 - naDataFormat_Fahrenheit

13 - naDataFormat_Centigrade

Notes:

- The [getData](#) (variant) method includes a "format" argument, which allows scalar (one-dimensional) data. To put data back into the "raw" data buffer using this (putDataComplex) method, specify **Polar** format when using the getData method.
- **Phase** format accepts data in radians (not degrees) and displays in degrees. To convert to degrees: radians * (57.29577951308233) = degrees. The getData method returns degrees if the request is for phase data.

data (**variant**) - A 1-dimension array of single precision floating point numbers.

Return Type Not Applicable

Default Not Applicable

Examples

```
' Put 201 points worth of scalar data into the measurement
' 200 is max index, so 0 to 200 is 201 points
Dim data(200) ' array of 201 (scalar) data points
' Fill the array
For i = 0 to 200
data(i) = i/200;
Next
app.ActiveMeasurement.putDataScalar 0, data
```

C++ Syntax HRESULT putDataScalar(tagNADataStore DataStore, VARIANT scalarArray)

Interface IMeasurement

Last Modified:

1-Oct-2007 Added temperature formats

PutShortcut Method

Description	Defines a Macro (shortcut) file in the analyzer. This command links a file name and path to the Macro file. The file must be put in the PNA at the location indicated by this command.
VB Syntax	<i>app</i> .PutShortcut <i>index,title,path, arguments</i>
Variable	(Type) - Description
<i>app</i>	An Application (object)
<i>index</i>	(long) - Number of the macro to be stored in the analyzer. Use a number between 1 and 25 . If the index number already exists, the existing macro is replaced with the new macro.
<i>title</i>	(string) - The name to be assigned to the macro
<i>path</i>	(string) - Full path, file name, and extension of the existing macro "executable" file.
<i>arguments</i>	(string) - Arguments that may be required for the specified macro to run.
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>app.PutShortcut 1,"Test","C:/Automation/MyTest.vbs", ""</code>
C++ Syntax	HRESULT PutShortcut(long Number, BSTR title, BSTR pathname,BSTR arguments)
Interface	IApplication

Last Modified:

29-Apr-2013 Updated total to 25
25-Jun-2009 Added arguments

putSourcePowerCalDataEx Method

Description	<p>Note: This method replaces putSourcePowerCalData Method</p> <p>Inputs source power calibration data (as variant data type) to this channel for a specific source port.</p> <p>The effect from this command on the channel is immediate. Do NOT send <code>ApplyPowerCorrectionValuesEX</code> after this command as it may invalidate the uploaded data.</p> <p>If the channel is sweeping the source backwards, then the first data point is the highest frequency value; the last data point is the lowest. Use the Get X-Axis Values command to return the X-axis values in the displayed order.</p> <p>The calibration is not valid if the current number of points on the channel is not equal to the number of values that were input.</p> <p>Note: This method sends variant data which is less efficient than methods available on the <code>ISourcePowerCalData</code> interface.</p>
VB Syntax	<code>chan.putSourcePowerCalDataEx buffer, srcPort, data</code>
Variable	(Type) - Description
<i>chan</i>	(object) – A Channel object
<i>buffer</i>	(enum <code>NASourcePowerCalBuffer</code>) - The source power cal data buffer to write to.
	0 - <code>naCorrectionValues</code> This is the only data buffer currently available.
<i>srcPort</i>	(long integer) – The source port for which calibration data is being requested.
	Note: If the port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-x model, then you must use chan.getPortNumber to translate the string into a port number. To learn more see Remotely Specifying a Source Port .
<i>data</i>	(variant) – Array of source power cal data being input.
Return Type	None
Default	Not Applicable
Examples	<code>chan.putSourcePowerCalDataEx naCorrectionValues, 1, varData</code>

C++ Syntax HRESULT putSourcePowerCalDataEx(tagNASourcePowerCalBuffer
bufSelect, long sourcePort, VARIANT varData);

Interface IChannel4

Last Modified:

25-Oct-2010	Added immediate note
24-Apr-2008	Added string names note
27-Jun-2007	Updated for PNA-X source port names

putSourcePowerCalDataScalarEx Method

Description	<p>Note: This method replaces putSourcePowerCalDataScalar Method</p> <p>Inputs source power calibration data (as scalar values) to this channel for a specific source port.</p> <p>The effect from this command on the channel is immediate. Do NOT send ApplyPowerCorrectionValuesEX after this command as it may invalidate the uploaded data.</p> <p>If the channel is sweeping the source backwards, then the first data point is the highest frequency value; the last data point is the lowest. Use the Get X-Axis Values2 command to return the X-axis values in the displayed order.</p>
VB Syntax	<i>chanData</i> .putSourcePowerCalDataScalarEx <i>buffer</i> , <i>srcPort</i> , <i>numValues</i> , <i>data</i>
Variable	(Type) - Description
<i>chanData</i>	(interface) – An ISourcePowerCalData2 interface on the Channel (object)
<i>buffer</i>	(enum NASourcePowerCalBuffer) - The source power cal data buffer to write to.
	0 - naCorrectionValues This is the only buffer currently available.
<i>srcPort</i>	(long integer) – The source port for which calibration data is being input.
	Note: If the port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-x model, then you must use chan.getPortNumber to translate the string into a port number. To learn more see Remotely Specifying a Source Port .
<i>numValues</i>	(long integer) – Number of data values being input.
	Note: If this does not equal the current number of points on the channel, the calibration will not be valid.
<i>data</i>	(single) – Array of source power cal data being input.
Return Type	None
Default	Not Applicable

Examples

```
Dim chanData As ISourcePowerCalData2
Set chanData = app.ActiveChannel
chanData.putSourcePowerCalDataScalarEx naCorrectionValues, 1,
201, scalarCalValues(0)
```

C++ Syntax

```
HRESULT putSourcePowerCalDataScalarEx(tagNASourcePowerCalBuffer
bufSelect, long sourcePort, long numValues, float *pData);
```

Interface

ISourcePowerCalData2

Last Modified:

25-Oct-2010 Added immediate note
24-Apr-2008 Added string name note

PutStandard Method **Superseded**

Description This command is [replaced](#) with [PutStandardByString](#)

Puts standard acquisition data into the Cal Set.

Learn more about [Reading and Writing Cal Data](#)

See examples of [Reading](#) and [Writing](#) Cal Set Data.

VB Syntax *CalSet.putStandard class, rcv, src, data*

Variable **(Type) - Description**

CalSet **(object)** - A [Cal Set](#) object

class **(enum NACalClass)** Standard. Choose from:

- 1 - naClassA
- 2 - naClassB
- 3 - naClassC
- 4 - naClassD
- 5 - naClassE
- 6 - naReferenceRatioLine
- 7 - naReferenceRatioThru

SOLT Standards

- 1 - naSOLT_Open
- 2 - naSOLT_Short
- 3 - naSOLT_Load
- 4 - naSOLT_Thru
- 5 - naSOLT_Isolation

TRL Standards

- 1 - naTRL_Reflection
- 2 - naTRL_Line_Reflection
- 3 - naTRL_Line_Tracking
- 4 - naTRL_Thru
- 5 - naTRL_Isolation

rcv **(long)** - Receiver Port

src (long) - Source Port

data (**variant**) Error term data in a two-dimensional array (0:1, 0:numpts-1).
The data must be complex pairs.

Note: See also [Put Standard Complex](#) on the ICalData2 interface to avoid using the variant data type.

Return Type Not Applicable

Default Not Applicable

Examples [See an Example](#)

C++ Syntax HRESULT putStandard(tagNACalClass stdclass, long ReceivePort, long SourcePort, VARIANT varData)

Interface ICalSet

PutStandardByString

Description Puts standard acquisition data into the Cal Set.

Learn more about [Reading and Writing Cal Data](#)

See examples of [Reading](#) and [Writing](#) Cal Set Data.

VB Syntax **PutStandardByString**(*stdName*, *vdata*)

Variable [\(Type\)](#) - Description

stdName **(String)** The string used to identify a particular standard in the Cal Set. An example string requesting the data for the Load standard in a full 2 port cal might be "S11C(3,3)".

vdata **(Variant)** The variant containing a safearray of variants. This data is usually two dimensional.

Note: The vardata array is a safearray of variants wrapped in a variant. This structure is compatible with scripting clients who can only use variants. For alternative methods that used typed arrays, see ICalData3.

Return Type Not Applicable

Default Not Applicable

Examples [See an Example](#)

C++ Syntax HRESULT PutStandardByString(BSTR bufferName, VARIANT vardata);

Interface ICalSet2

PutStandardComplex Method **Superseded**

Description This command is replaced with [PutStandardComplexByString](#)

Puts standards acquisition data into the Cal Set.

Learn more about [Reading and Writing Cal Data](#)

See examples of [Reading](#) and [Writing](#) Cal Set Data

VB Syntax *ICalData2*.**putStandardComplex** *class, rcv, src, numPts, real(), imag()*

Variable **(Type) - Description**

ICalData2 An [ICalData2](#) pointer to the Cal Set object

class (**enum NA CalClass**) Standard. Choose from:

- 1 - naClassA
- 2 - naClassB
- 3 - naClassC
- 4 - naClassD
- 5 - naClassE
- 6 - naReferenceRatioLine
- 7 - naReferenceRatioThru

SOLT Standards

- 1 - naSOLT_Open
- 2 - naSOLT_Short
- 3 - naSOLT_Load
- 4 - naSOLT_Thru
- 5 - naSOLT_Isolation

TRL Standards

- 1 - naTRL_Reflection
- 2 - naTRL_Line_Reflection
- 3 - naTRL_Line_Tracking
- 4 - naTRL_Thru
- 5 - naTRL_Isolation

rcv (**long**) - Receiver Port

<i>src</i>	(long) - Source Port
<i>numPts</i>	(long) - The number of data points in the real and imaginary arrays.
<i>real()</i>	(single) - one-dimensional array containing the real part of the acquisition data. (0:points-1)
<i>imag()</i>	(single) - one-dimensional array containing the imaginary part of the acquisition data. (0:points-1)
Return Type	Not Applicable
Default	Not Applicable
Examples	<pre>Dim sdata As ICalData2 Set sdata = calmanager.CreateCal Set(1) sdata.putStandardComplex naSOLT_Open, 1, 1, numpts, rel(0), img(0)</pre>
<u>C++ Syntax</u>	HRESULT putStandardComplex(tagNACalClass stdclass, long ReceivePort, long SourcePort, long INumValues, float* pReal, float* pImag)
Interface	ICalData2

PutStandardComplexByString

Description	Puts standard acquisition data into the Cal Set. Learn more about Reading and Writing Cal Data See examples of Reading and Writing Cal Set Data.
VB Syntax	<code>ICalData3.PutStandardComplexByString(stdName, lnumPoints , real(o) , imag(0))</code>
Variable	(Type) - Description
<i>ICalData3</i>	An ICalData3 pointer to a Cal Set object.
<i>stdName</i>	(String) The string used to identify a particular standard in the Cal Set. An example string requesting the data for the Load standard in a full 2 port cal might be "S11C(3,3)".
<i>lnumpoints</i>	(long) - The number of data points in the real and imaginary arrays.
<i>real</i>	(Single) The real component of the complex data.
<i>imag</i>	(Single) The imaginary component of the complex data.
Return Value	Single
Default	Not Applicable
Examples	See an Example
C++ Syntax	<code>HRESULT PutStandardComplexByString(BSTR bufferName, long lnumPoints, float* real, float* imag);</code>
Interface	ICalData3

Quit Method

Description Terminates the Network Analyzer application.

VB Syntax *app*.Quit

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

Return Type Not Applicable

Default Not Applicable

Examples `app.Quit`

C++ Syntax HRESULT Quit()

Interface IApplication

Remarks Under the rules of COM, the server should not exit until all references to it have been released. This method is a brute force way of terminating the application. Be sure to release all references (or terminate the client program) before attempting to restart the Network Analyzer application.

An alternate approach to terminating the application is to make the application invisible (`app.Visible = False`) and release all references. The server will shutdown.

ReadData Method

Description	Reads a 13-bit data word from the specified address. Data is read using the AD0 through AD12 lines of the external test set connector. The instrument generates the appropriate timing signals. It automatically controls timing signals LDS, LAS and RLW to strobe the address, and then read the data, from the external test set. See the timing diagram for Address and Data I/O read.
VB Syntax	<i>value</i> = <i>ExtIO</i> . ReadData (<i>address</i>)
Variable	(Type) - Description
<i>value</i>	(variant) - Variable to store the returned data
<i>ExtIO</i>	(object) - An ExternalTestSetIO object
<i>address</i>	(variant) - address to read data from.
Return Type	Variant
Default	Not Applicable
Examples	<pre>value = ExtIO.ReadData (15)</pre>
C++ Syntax	HRESULT ReadData (VARIANT Address, VARIANT* Data);
Interface	IHWExternalTestSetIO

ReadRaw Method

Description Reads a 16-bit value from the external test set. The 16-bit value is comprised of lines AD0 - AD12, Sweep Holdoff In and Interrupt In (inverted).

When this command is used the analyzer does NOT generate the appropriate timing signals; it simply reads the lines. The user needs to first use the [WriteRaw](#) method to do the initial setup. The RLW line (pin25) must be set to the appropriate level in order to read the test set connected.

Below is the format of data that is read with ReadRaw:

Pin	Bit	Signal name
22	0	AD0*
23	1	AD1*
11	2	AD2*
10	3	AD3*
9	4	AD4*
21	5	AD5*
20	6	AD6*
19	7	AD7*
6	8	AD8*
5	9	AD9*
4	10	AD10*
17	11	AD11*
3	12	AD12*
2	13	Sweep Holdoff In
13	14	Interrupt In (inverted internally)
na	15	Always Zero, grounded internally

*These lines are dependent on the state of RLW (pin25).
Writing a 0(low) to RLW will set lines AD0-AD12 to write mode.
Writing a 1(high) to RLW will set lines AD0-AD12 to read mode.

VB Syntax *value* = **ExtIO.ReadRaw** (*address*)

Variable [\(Type\)](#) - Description

value **(variant)** - Variable to store the returned data

ExtIO **(object)** - An External IO object

address **(variant)** - Address to read data from

Return Type Real

Default Not Applicable

Examples `value = ExtIO.ReadRaw (address)`

C++ Syntax HRESULT ReadRaw(VARIANT* Input);

Interface IHWExternalTestSetIO

ReCalculate Method

Description	Repeats the last calculation that was performed, including all ON (state) segments in segment table.
VB Syntax	<i>conv</i> . ReCalculate ()
Variable	(Type) - Description
<i>conv</i>	A Converter Object
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>mxr.ReCalculate()</code>
	See example program
C++ Syntax	HRESULT ReCalculate()
Interface	IConverter5

Last Modified:

26-Oct-2010 New command (A.09.33)

Recall Method

Description Recalls a measurement state, calibration state, or both, from the hard drive into the analyzer.

Use `app.Save` to save files.

VB Syntax `app.Recall (filename.ext)`

Variable **(Type) - Description**

`app` An [Application](#) (object)

`filename.ext` **(string)** - Full path, file name, and extension, of the file.

Files are typically stored in "c:\users\public\network analyzer\documents"

Use one of the following extensions:

- .sta - Instrument State
- .cal - Calibration file
- .cst - Both Instrument State and Calibration reference
- .cti - Citifile (data will always be formatted. See [Recalling Citifiles Using the PNA](#))
- .csa - Instrument state and calibration data (not a reference pointer).

Return Type Not Applicable

Default Not Applicable

Examples `app.Recall ("C:/Program Files/Agilent/Network Analyzer/Documents/MyState.cst")` 'Recalls "mystate.cst" from the specified folder

C++ Syntax HRESULT Recall(BSTR bstrFile)

Interface IApplication

Recall Kits Method

Description	Recalls the calibration kits definitions that were stored with the SaveKits command.
VB Syntax	<i>app</i> . RecallKits
Variable	(Type) - Description
<i>app</i>	An Application (object)
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>app.RecallKits</code>
C++ Syntax	HRESULT RecallKits()
Interface	IApplication

Remove Method

Description Removes an item from a collection of objects.

VB Syntax *Object*.**Remove** *item*

Variable [\(Type\)](#) - Description

Object Any of the following (**objects**)

- [CalFactorSegments collection](#)
- [CalFactorSegmentsPMAR Collection](#)
- [Cal Sets collection](#)
- [Channels Collection](#)
- [ExternalDevices Collection](#)
- [GuidedCalibrationPowerSensors Collection](#)
- [Measurements collection](#)
- [NAWindows collection](#)
- [PowerLossSegments collection](#)
- [PowerLossSegmentsPMAR_Collection](#)
- [Segments collection](#)

Note: Segments, CalFactorSegments, and PowerLossSegments have an OPTIONAL argument [size] referring to the number of segments to remove, starting with the *item* parameter.

Note: Segments - When ALL segments are deleted, [SweepType](#) is automatically set to Linear because there are no segments to sweep.

item (**variant**) - Collection Item number to be removed.

Note: The ExternalDevices Collection requires that you specify *item* as the string name of the device. For example:

```
extDevices.Remove ('mySource')
```

Return Type Not Applicable

Default Not Applicable

Examples

```
Measurements.Remove 3 'Removes the third measurement in the collection
```

```
segments.Remove 2,20 'Removes 20 segments (2 - 21)
```

C++ Syntax

```
HRESULT Remove(VARIANT index); //Measurements
```

```
HRESULT Remove(VARIANT index); //Cal Sets
```

```
HRESULT Remove(long windowNumber); //NAWindows
```

```
HRESULT Remove(VARIANT index, long size); //Segments
```

```
HRESULT Remove(VARIANT index, long size);  
//CalFactorSegments(PMAR)
```

```
HRESULT Remove(VARIANT index, long size);  
//PowerLossSegments(PMAR)
```

```
HRESULT Remove(BSTR name) //ExternalDevices
```

```
HRESULT Remove( VARIANT index) // Channels - specify collections index,  
not the channel number.
```

```
HRESULT Remove(VARIANT index); //GuidedCalibrationPowerSensors
```

Interface

All listed above

Last Modified:

8-Feb-2011	Added GuidedCalibrationPowerSensors (9.33)
16-Sep-2010	Added channels (9.30)
31-Jul-2009	Added External Devices (9.0)

Write-only

RemoveAll Method

Description Removes ALL power sensors from the [GuidedCalibrationPowerSensors Collection](#)

VB Syntax *guidedSensors*.**RemoveAll**

Variable [\(Type\)](#) - Description

guidedSensors A [GuidedCalibrationPowerSensors](#) Collection

Return Type Not Applicable

Default Not Applicable

Examples `sensors.RemoveAll`

C++ Syntax HRESULT RemoveAll();

Interface IGuidedCalibrationPowerSensors

Last Modified:

8-Feb-2011 New topic (9.33)

RemoveChannelNumber Method

Description	Deletes a channel by specifying the channel number. Use Remove Method to delete a channel by specifying the index in the channels collection.
VB Syntax	<i>chans</i> . RemoveChannelNumber (<i>chan</i>)
Variable	(Type) - Description
	<i>chans</i> A Channels (collection)
	(<i>chan</i>) The channel number to delete.
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>chan.RemoveChannelNumber (2)</code>
C++ Syntax	HRESULT RemoveChannelNumber(VARIANT channelNumber)
Interface	IChannels3

Last Modified:

16-Sep-2010 MX New topic

RemoveItem Method

Description Removes a name-value pair from the Cal Set. Send the [Save \(CalSet\) Method](#) to save the edited CalSet to the PNA.

See Also

[EnumerateItems Method](#)

[Item Property](#) (Learn about name-value pairs)

VB Syntax *CalSet*.**RemoveItem** (*name*)

Variable [\(Type\)](#) - Description

CalSet **(object)** - A [CalSet](#) object

name (String) Name of the item.

Return Type Not Applicable

Default Not Applicable

Examples [See example](#)

C++ Syntax HRESULT RemoveItem (VARIANT* itemNames);

Interface ICalSet6

Last Modified:

24-Sep-2010 MX New topic

RemoveLibrary Method

Description Removes an imported an Equation Editor DLL from the PNA.

VB Syntax *equation.RemoveLibrary location*

Variable [\(Type\)](#) - Description

equation A [MeasurementEquation](#) object

location **(string)** – Full path and filename of the *.dll to be removed.

Return Type Not Applicable

Default Not Applicable

Examples `equation.RemoveLibrary "C:/Program Files/Agilent/Network Analyzer/UserFunctions/Expansion.dll"`

C++ Syntax HRESULT RemoveLibrary(BSTR filename);

Interface IMeasurementEquation2

Last Modified:

10-Jan-2011 New topic

Reset Method

Description	Removes all existing windows and measurements from the application. (Unlike Preset , does not create a new measurement.)
VB Syntax	<i>app</i> .Reset
Variable	(Type) - Description
<i>app</i>	An Application (object)
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>app.Reset</code>
C++ Syntax	HRESULT Reset()
Interface	IApplication

Reset Method (Cal All)

Description	Resets all properties associated with the Cal All session to their default values.
VB Syntax	<i>calAll</i> .Reset
Variable	(Type) - Description
<i>calAll</i>	A CalibrateAllChannels (object)
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>calAll.Reset</code>
C++ Syntax	HRESULT Reset()
Interface	ICalibrateAllChannels

Last modified:

4-Jan-2012 New topic

Reset Method (PhaseRef)

Description	Resets all properties associated with the Phase Reference Cal to their default values. For predictable results, send this command before performing each Phase Reference Cal.
VB Syntax	<i>phaseRef</i> . Reset
Variable	(Type) - Description
<i>phaseRef</i>	A PhaseReferenceCalibration (object)
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>phaseRef.Reset ()</code>
C++ Syntax	HRESULT Reset()
Interface	IPhaseReferenceCalibration

Last modified:

11-Mar-2012 New topic

ResetLOFrequency Method

Description Resets the LO Delta Frequency to 0 (zero) Hz.

VB Syntax *obj*.ResetLOFrequency

Variable [\(Type\)](#) - Description

obj An [EmbeddedLO](#) (**object**) or

A [ConverterEmbeddedLO](#) (**object**)

Default Not Applicable

Examples `embedLO.ResetLOFrequency 'write`

C++ Syntax HRESULT ResetLOFrequency();

Interface IEmbeddedLO

Last Modified:

12-Aug-2009 Added ConvEmbedLO object

18-Apr-2007 MX New topic

ResetTheme Method

Description Resets the current theme to the default PNA colors.

VB Syntax *colors*.**ResetTheme**()

Variable [\(Type\)](#) - Description

colors A [ComColors](#) (**object**)

Return Type Not Applicable

Default Not Applicable

Examples `colors.ResetTheme()`

C++ Syntax HRESULT ResetTheme();

Interface IComColors

Last Modified:

7-Aug-2009 MX New topic

ResetTuningParameters Method

Description Resets the tuning parameters to their default values.

VB Syntax *obj*.ResetTuningParameters

Variable [\(Type\)](#) - Description

obj An [EmbeddedLO](#) (**object**) or

A [ConverterEmbeddedLO](#) (**object**)

Default Not Applicable

Examples `embedLO.ResetTuningParameters 'write`

C++ Syntax HRESULT ResetTuningParameters();

Interface IEmbeddedLO

Last Modified:

12-Aug-2009 Added ConvEmbedLO object

18-Apr-2007 MX New topic

RestoreCalKitDefaults Method

Description Restores the original properties of the specified Cal Kit, overwriting the last definition with the factory defaults.

NOTE: ONLY works with PNA releases 1.0 through 1.6.

VB Syntax `app.RestoreCalKitDefaults (calKit)`

Variable **(Type) - Description**

app An [Application](#) (object)

calKit (enum NACalKit) - Calibration Kit to restore. Choose from:

- 1 - naCalKit_85032F_N50
- 2 - naCalKit_85033E_3_5
- 3 - naCalKit_85032B_N50
- 4 - naCalKit_85033D_3_5
- 5 - naCalKit_85038A_7_16
- 6 - naCalKit_85052C_3_5_TRL
- 7 - naCalKit_User7
- 8 - naCalKit_User8
- 9 - naCalKit_User9
- 10 - naCalKit_User10

Return Type Not Applicable

Default Not Applicable

Examples `app.RestoreCalKitDefaults naCalKit_MechKit10`

C++ Syntax HRESULT RestoreCalKitDefaults/(tagNACalKit kit)

Interface IApplication

RestoreCalKitDefaultsAll Method

Description Restores the original properties of ALL of the Cal Kits, overwriting the last definitions with the factory defaults.

NOTE: ONLY works with PNA releases 1.0 through 1.6.

VB Syntax *app*.RestoreCalKitDefaultsAll

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

Return Type Not Applicable

Default Not Applicable

Examples `app.RestoreCalKitDefaultsAll`

C++ Syntax HRESULT RestoreCalKitDefaultsAll()

Interface IApplication

RestoreDefaults Method

Description Resets the PNA preferences to their factory default settings.

VB Syntax *pref*.RestoreDefaults

Variable [\(Type\)](#) - Description

pref A [Preferences](#) (object)

Return Type Not Applicable

Default Not Applicable

Examples `pref.RestoreDefaults`

C++ Syntax HRESULT RestoreDefaults()

Interface IPreferences9

Last Modified:

24-Apr-2008 MX New topic

Resume Method

Description	Resumes the trigger mode of all channels that was in effect before sending the channels.Hold method. Channels.Hold must be sent before channels.Resume, using the same instance of the Channels object.
VB Syntax	<i>chans</i> .Resume
Variable	(Type) - Description
	<i>chans</i> A Channel collection (object)
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>chans.Resume</code>
C++ Syntax	HRESULT Resume();
Interface	IChannels2

Save Method

Description Saves the appropriate content to the hard drive depending on the extension that is provided.

Some saved files can be recalled using app.[Recall](#). depending on the content.

VB Syntax *app*.**Save**(*filename.ext*)

Variable **(Type) - Description**

app An [Application](#) (**object**)

filename.ext (**string**) - Full path, file name, and extension of the file.

Files are typically stored in "c:\users\public\network analyzer\documents"

Use one of the following extensions:

- **.cst** - Saves both Instrument State and Cal Set reference - Recalls a calibrated measurement. (Recallable)
- **.sta** - Saves Instrument State only - recalls the instrument state without calibration. (Recallable)
- **.cal** - Calibration file – saves the active Cal Sets currently in use by any channel. Use this mode for archival purposes only. All Cal Sets are saved to a Cal Set data file. This mode provides a method of safeguarding calibration data. This data can be restored to the list of Cal Sets available in the instrument. (Recallable)
- **.csa** - Saves both instrument state AND actual calibration data, not a reference pointer to the Cal Set.
- **.prn** - Saves active trace in comma-separated format (not recallable)
- **.bmp** - Saves a Bitmap of the screen (not recallable)
- **.s1p** - Saves 1-port measurement data
- **.s2p** - Saves 2-port measurement data
- **.s3p** - Saves 3-port measurement data
- **.s4p** - Saves 4-port measurement data

Return Type Not Applicable

Default Not Applicable

Examples

```
app.Save("C:/Program Files/Agilent/Network  
Analyzer/Documents/Newfolder/MyState.cst") 'Saves "mystate.cst"  
to the specified folder
```

C++ Syntax

HRESULT Save(BSTR bstrFile)

Interface

IApplication

Last Modified:

26-Jun-2007 Corrected example

Save Method

Description Saves the current Cal Set to disk. This is the recommended method for saving a Cal Set.

Learn more about [reading and writing Cal data using COM](#)

VB Syntax *CalSet*.Save

Variable [\(Type\)](#) - Description

CalSet **(object)** - A [CalSet](#) object

Return Type Not Applicable

Default Not Applicable

Examples `myCalSet.Save`

See [Copy Method](#) for an example application of this command.

C++ Syntax HRESULT Save();

Interface ICalSet

SaveCalSets Method **Superseded**

Description This command is replaced by [ICalSet::Save](#) which saves the data for **only** the current Cal Set to the disk.

Writes new or changed Cal Sets to disk. All Cal Sets are saved in a single file. This file is updated at the following times:

- When a Cal Set has been deleted.
- When a calibration has been performed through the front panel interface.
- When this method is called.
- When [ICalSet::Save](#) is called.

Learn more about [reading and writing Cal data using COM](#)

VB Syntax *object*.SaveCalSets

Variable [\(Type\)](#) - Description

object **(object)** - A CalManager object or a Calibrator object

Return Type None

Default Not Applicable

Example `calMgr.SaveCalSets`

C++ Syntax HRESULT SaveCalSets();

Interface ICalManager
ICalibrator

SaveCitiDataData Method - Superseded

Description	This command is replaced with SaveData Method Saves UNFORMATTED trace data to .cti file. Learn more about citifiles.
VB Syntax	<code>app.SaveCitiDataData(filename.cti)</code>
Variable	(Type) - Description
<i>app</i>	An Application (object)
<i>filename.cti</i>	(string) - Full path, file name, and .cti extension of the file. Files are typically stored in "c:\users\public\network analyzer\documents".
Return Type	Not Applicable
Default	Not Applicable
Examples	<pre>app.SaveCitiDataData ("C:/Program Files/Agilent/Network Analyzer/Documents/myDDCitifile.cti") 'Saves "myDDCitifile.cti" to the specified folder</pre>
C++ Syntax	HRESULT SaveCitiDataData (BSTR bstrFile)
Interface	IApplication5

SaveCitiFormattedData Method - Superseded

Description	This command is replaced with SaveData Method Saves FORMATTED trace data to .cti file. Learn more about citifiles.
VB Syntax	<i>app</i> .SaveCitiFormattedData(<i>filename.cti</i>)
Variable	(Type) - Description
<i>app</i>	An Application (object)
<i>filename.cti</i>	(string) - Full path, file name, and .cti extension of the file. Files are typically stored in "c:\users\public\network analyzer\documents"
Return Type	Not Applicable
Default	Not Applicable
Examples	<pre>app.SaveCitiFormattedData ("C:/Program Files/Agilent/Network Analyzer/Documents/Newfolder/myFDCitifile.cti") 'Saves "myFDCitifile.cti" to the specified folder</pre>
C++ Syntax	HRESULT SaveCitiFormattedData (BSTR bstrFile)
Interface	IApplication5

SaveData Method

Description Stores trace data to the following file types: *.prn, *.cti, *.csv, *.mdf

To save snp files, use [WriteSnpFileWithSpecifiedPorts](#)

To save instrument state and calibration files, use [Save](#).

This command replaces the following:

[SaveCitiDataData Method](#)

[SaveCitiFormattedData Method](#)

[CitiContents Property](#)

[CitiFormat Property](#)

Some saved files can be recalled using app.[Recall](#). depending on the content.

VB Syntax `app.SaveData filename,type,scope,format,selector`

Variable **(Type) - Description**

app An [Application](#) (**object**)

filename (**string**) - Full path, file name, and extension of the file.

Files are typically stored in "c:\users\public\network analyzer\documents"

all other parameters Choose from the following valid parameter combinations for ALL measurement classes:

	Parameters			
Type of file to save	<type> (String)	<scope> (String)	<format> (String)	<selector> (Integer)
Click to learn about these file types:				
*.prn	"PRN Trace Data"	"Trace"	"Displayed"	Measurement number
Example: <code>app.SaveData "myData.prn","PRN Trace Data","Trace","Displayed",2</code>				

*.cti (unformatted)	"Citifile Data Data"	"Trace" or "Auto"	"RI"	Measurement number
--	-------------------------	----------------------	------	-----------------------

Example: app.SaveData "myData.cti","Citifile Data Data","AUTO","RI",3

*.cti (unformatted)	"Citifile Data Data"	"Displayed"	"RI"	-1
--	-------------------------	-------------	------	----

Example: app.SaveData "myData.cti","Citifile Data Data","AUTO","RI",3

*.cti (formatted)	"Citifile Formatted Data"	"Trace" or "Auto"	"RI" or "MA" or "DB"	Measurement number
--------------------------------------	---------------------------------	----------------------	-------------------------	-----------------------

Example: app.SaveData "myData.cti","Citifile Formatted
Data","AUTO","MA",3

*.cti (formatted)	"Citifile Formatted Data"	"Displayed"	"RI" or "MA" or "DB" or "Displayed"	-1
--------------------------------------	---------------------------------	-------------	---	----

Example: app.SaveData "myData.cti","Citifile Formatted
Data","DISPLAYED","MA",-1

*.csv	"CSV Formatted Data"	"Trace" or "Auto"	"RI" or "MA" or "DB" or "Displayed"	Measurement number
-----------------------	----------------------------	----------------------	---	-----------------------

Example: app.SaveData "myData.csv","CSV Formatted
Data","Trace","DB",3

*.csv	"CSV Formatted Data"	"Displayed"	"RI" or "MA" or "DB"	-1
-----------------------	----------------------------	-------------	-------------------------	----

Example: app.SaveData "myData.csv","CSV Formatted
Data","displayed","RI",-1

*.mdf	"MDIF Data"	"Trace" or "Auto"	"RI" or "Displayed"	Measurement number
-----------------------	-------------	----------------------	------------------------	-----------------------

Example: app.SaveData "myData.mdf","MDIFData","trace","displayed",1

*.mdf	"MDIF Data"	"Displayed"	"RI" or "Displayed"	-1
-----------------------	-------------	-------------	------------------------	----

Example: app.SaveData "myData.mdf","MDIF
Data","displayed","displayed",-1

Notes (for above file types)

Use [meas.Number](#) to read the measurement number of a trace.

Scope:

"Trace" - specified measurement number only.

"Displayed" - all displayed measurements.

"Auto" - for all Standard Meas Class (S-parameter) channels:

- When correction is OFF, saves the specified trace
- When correction is ON, saves all corrected parameters associated with the calibrated ports in the Cal Set.

"Auto" - for all other channels:

- When correction is OFF or ON, saves the specified trace

The following parameter combinations save *.csv files in specific formats for GCA and Swept IMD classes:

	Parameters			
	<type> (String)	<scope> (String)	<format> (String)	<selector> (Integer)
GCA channels ONLY:	"GCA Sweep Data"	"Auto"	"DB"	GCA channel number

[Learn more](#)

Example: `app.SaveData "myData.csv","GCA Sweep Data","Auto","db",1`

Swept IMD channels ONLY:	"IMD Sweep Data"	"Auto"	"DB"	Swept IMD channel number
--------------------------	------------------	--------	------	--------------------------

[Learn more](#)

Example: `app.SaveData "myData.csv","IMD Sweep Data","Auto","db",1`

Return Type Not Applicable

Default Not Applicable

[C++ Syntax](#) HRESULT SaveData(BSTR File, BSTR Type, BSTR Scope, BSTR Format,
Long selector);

Interface IApplication18

Last Modified:

9-Apr-2010 MX New topic

SaveENRFile Method

Description	Saves an ENR table to disk.
VB Syntax	<i>enr</i> . SaveENRFile (<i>filename</i>)
Variable	(Type) - Description
<i>enr</i>	An ENRFile (object)
<i>filename</i>	(String) - Absolute path and filename of the ENR file.
Return Type	Not Applicable
Default	Not Applicable
Examples	See example program
C++ Syntax	HRESULT SaveENRFile(BSTR filename);
Interface	IENRFile

Last Modified:

2-Aug-2007 MX New topic

Write only

SaveFile Method

Description Saves the mixer/converter test setup to a mixer attributes (.mxr) file.

VB Syntax *obj*.**SaveFile** (*filename*)

Variable [\(Type\)](#) - Description

obj A [Mixer Interface](#) pointer to the [Measurement](#) (object)

Or

A [Converter Object](#)

filename (String) Full path, file name, and .mxr extension of the file.

Files are typically stored in "c:\users\public\network analyzer\documents".

Return Type String

Default Not Applicable

Examples `mixer.SaveFile ("c:\users\public\network analyzer\documents/myMixer.mxr")`

C++ Syntax HRESULT SaveFile(BSTR newVal)

Interface IMixer

IConverter

Last Modified:

2-Feb-2009 Added Converter

Write only

SaveFile Method

Description	Saves an external device configuration file to the PNA hard drive. Currently, only DC Supply and DC Meter configuration files are supported. See more DC Device commands .
VB Syntax	<i>extDev</i> . SaveFile (<i>filename</i>)
Variable	(Type) - Description
<i>extDev</i>	An ExternalDevice Object
<i>filename</i>	(String) File name and .xml extension of the external device configuration file. Currently, only DC Supply and DC Meter configuration files are supported. See more DC Device commands .
Return Type	Not Applicable
Default	Not Applicable
Examples	<pre>extDev.SaveFile "MyDCSupply"</pre>
C++ Syntax	HRESULT SaveFile(BSTR newVal)
Interface	IExternalDevice

Last Modified:

11-Apr-2012 New topic

SaveToDiskMemory Method

Description Saves the User Characterization to PNA disk memory. To save to ECal internal memory, use [SaveToECal Method](#). User Characterization can be saved to both PNA disk memory and ECal module memory.

Note: An ECal confidence check can NOT be performed remotely from User Characterizations that are stored on the PNA disk.

VB Syntax `userChar.SaveToDiskMemory (name)`

Variable [\(Type\)](#) - Description

userChar An [ECalUserCharacterizer](#) (object)

name (String) User characterization name. Although there is no limit to the number of characters, only about 10 characters appear in the Cal Wizard dialog when selecting a user characterization for use.

Return Type Not Applicable

Default Not Applicable

Examples `userChar.SaveToDiskMemory "DUT1"`

C++ Syntax HRESULT SaveToDiskMemory(BSTR name);

Interface IECalUserCharacterizer2

Last Modified:

15-Jun-2010 Added Note
25-Aug-2009 MX New topic

SaveToECal Method

Description	Saves the User Characterization to the ECal module. This can take several minutes to complete. To save to PNA disk memory, use SaveToDiskMemory Method . User Characterization can be saved to both PNA disk memory and ECal module memory.
VB Syntax	<i>userChar</i> . SaveToECal
Variable	(Type) - Description
<i>userChar</i>	An ECalUserCharacterizer (object)
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>userChar.SaveToECal</code>
C++ Syntax	HRESULT SaveToECal();
Interface	IECalUserCharacterizer

Last Modified:

2-Nov-2008 New topic (8.33)

SaveKits Method

Description	Saves the cal kits, typically after modifying a calibration kit. To load a cal kit into the analyzer from the hard drive, use app.RecallKits .
VB Syntax	<i>app</i> . SaveKits
Variable	(Type) - Description
<i>app</i>	An Application (object)
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>app.SaveKits</code>
C++ Syntax	HRESULT SaveKits()
Interface	IApplication

SearchCompressionPoint Method

Description Searches the markers domain for the specified [compression level](#).

VB Syntax *mkr*.SearchCompressionPoint

Variable [\(Type\)](#) - Description

mkr A [Marker](#) (**object**)

Return Type Not Applicable

Default Not Applicable

Examples `mkr.SearchCompressionPoint 'Read'`

[See example program](#)

C++ Syntax HRESULT SearchCompressionPoint()

Interface IMarker4

Last Modified:

8-Feb-2009 MX New topic

SearchPowerNormalOperatingPoint Method

Description	<p>Initiates a PNOP marker search.</p> <p>Turns on and sets markers 1, 2, 3, and 4 to calculate various PNOP parameters.</p> <p>First set BackOff and PinOffset.</p> <p>To turn off these markers, either turn them off individually or DeleteAllMarkers.</p> <p>To search a UserRange, first activate marker 1 and set the desired UserRange. Then send the SearchPowerNormalOperatingPoint command. The user range applies only to marker 1 searching for the max value. The other markers may fall outside the user range.</p>
VB Syntax	<code>pnop.SearchPowerNormalOperatingPoint()</code>
Variable	(Type) - Description
	<code>pnop</code> A PNOP (object)
Return Type	Not Applicable
Default	Not Applicable
Examples	<pre>pnop.SearchPowerNormalOperatingPoint</pre> <p>See example program</p>
C++ Syntax	<code>HRESULT SearchPowerNormalOperatingPoint()</code>
Interface	IPNOP

Last Modified:

19-Feb-2010 MX New topic

SearchPowerSaturation Method

Description	<p>Initiates a Power Saturation marker search.</p> <p>Turns on and sets markers 1, 2, and 3 to calculate various Power Saturation parameters.</p> <p>First set PMaxBackOff.</p> <p>To turn off the Power Saturation markers, either turn them off individually or use DeleteAllMarkers Method</p> <p>To search a User Range with the PSAT search, first activate marker 1 and set the desired User Range. Then send this command. The user range used with the PSAT search only applies to marker 1 searching for the linear gain value. The other markers may fall outside the user range.</p>
VB Syntax	<code>pSat.SearchPowerSaturation()</code>
Variable	(Type) - Description
	<code>pSat</code> A PSaturation (object)
Return Type	Not Applicable
Default	Not Applicable
Examples	<pre>pSat.SearchPowerSaturation</pre> <p>See example program</p>
C++ Syntax	HRESULT SearchPowerSaturation()
Interface	IPSaturation

Last Modified:

19-Feb-2010 MX New topic

SearchFilterBandwidth Method

Description Searches the measurement data with the current [BandwidthTarget](#) (default is -3). To continually track the filter bandwidth, use [BandwidthTracking](#).

This feature uses markers 1-4. If not already, they are activated. To turn off these markers, either turn them off individually or [DeleteAllMarkers](#).

The bandwidth statistics are displayed on the analyzer screen. To get the bandwidth statistics, use either [GetFilterStatistics](#) or [FilterBW](#), [FilterCF](#), [FilterLoss](#), or [FilterQ](#).

The analyzer screen will show either Bandwidth statistics OR Trace statistics; not both.

To search a [UserRange](#) with the bandwidth search, first activate marker 1 and set the desired UserRange. Then send the SearchFilterBandwidth command. The user range used with bandwidth search only applies to marker 1 searching for the max value. The other markers may fall outside the user range.

VB Syntax `meas.SearchFilterBandwidth`

Variable [\(Type\)](#) - Description

meas A Measurement (**object**)

Return Type Not Applicable

Default Not Applicable

Examples `meas.SearchFilterBandwidth`

C++ Syntax HRESULT SearchFilterBandwidth()

Interface IMeasurement

SearchMax Method

Description Searches the marker domain for the maximum value.

VB Syntax *mark*.SearchMax

Variable [\(Type\)](#) - Description

mark A [Marker](#) (**object**)

Return Type Not Applicable

Default Not Applicable

Examples `mark.SearchMax`

C++ Syntax HRESULT SearchMax()

Interface IMarker

SearchMin Method

Description Searches the marker domain for the minimum value.

VB Syntax *mark*.SearchMin

Variable [\(Type\)](#) - Description

mark A [Marker](#) (**object**)

Return Type Not Applicable

Default Not Applicable

Examples `mark.SearchMin`

C++ Syntax HRESULT SearchMin()

Interface IMarker

SearchNextPeak Method

Description Searches the marker's domain for the next peak value.

VB Syntax *mark*.SearchNextPeak

Variable [\(Type\)](#) - Description

mark A [Marker](#) (**object**)

Return Type Not Applicable

Default Not Applicable

Examples `mark.SearchNextPeak`

C++ Syntax HRESULT SearchNextPeak()

Interface IMarker

SearchPeakLeft Method

Description	Searches the marker's domain for the next VALID peak to the left of the marker.
VB Syntax	<i>mark</i> .SearchPeakLeft
Variable	(Type) - Description
<i>mark</i>	A Marker (object)
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>mark.SearchPeakLeft</code>
C++ Syntax	HRESULT SearchPeakLeft()
Interface	IMarker

SearchPeakRight Method

Description	Searches the marker's domain for the next VALID peak to the right of the marker.
VB Syntax	<code>mark.SearchPeakRight</code>
Variable	(Type) - Description
<code>mark</code>	A Marker (object)
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>mark.SearchPeakRight</code>
C++ Syntax	HRESULT SearchPeakRight()
Interface	IMarker

SearchTarget Method

Description	Searches the marker's domain for the target value (specified with mark.TargetValue). Searches to the right; then at the end of the search domain, begins again at the start of the search domain.
VB Syntax	<code>mark.SearchTarget</code>
Variable	(Type) - Description
<code>mark</code>	A Marker (object)
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>mark.SearchTarget</code>
C++ Syntax	HRESULT SearchTarget()
Interface	IMarker

SearchTargetLeft Method

Description	Moving to the left of the marker position, searches the marker's domain for the target value (specified with mark.TargetValue).
VB Syntax	<i>mark</i> .SearchTargetLeft
Variable	(Type) - Description
<i>mark</i>	A Marker (object)
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>mark.SearchTargetLeft</code>
C++ Syntax	HRESULT SearchTargetLeft()
Interface	IMarker

SearchTargetRight Method

Description	Moving to the right of the marker position, searches the marker's domain for the target value (specified with mark.TargetValue).
VB Syntax	<code>mark.SearchTargetRight</code>
Variable	(Type) - Description
<i>mark</i>	A Marker (object)
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>mark.SearchTargetRight</code>
C++ Syntax	HRESULT SearchTargetRight()
Interface	IMarker

SegmentCalculate Method

Description Calculates the specified parameter for the segment.

VB Syntax *conv*.**SegmentCalculate** *index,param*

Variable **(Type) - Description**

conv A [Converter Object](#)

index (Long integer) Segment for which calculation is performed. Choose a segment between 1 and the current segment count. Use [SegmentCount Property](#) to read the current count in the [Applied Mixer](#).

param (Enum as ConverterCalculation) Mixer port for which to calculate start and stop frequencies. Choose from:

enum	1st or only stage requires:	In addition, 2nd stage requires:
0 naCalculateINPUT	<ul style="list-style-type: none"> Output Start and Stop frequencies LO frequency Output sideband (High or Low) 	<ul style="list-style-type: none"> IF Start and Stop frequencies 2nd LO frequency IF sideband (High or Low)
1 naCalculateINPUT AndOUTPUT (2 stage mixers ONLY)	NA	<ul style="list-style-type: none"> IF Start and Stop frequencies Both LO frequencies
2 naCalculateOUTPUT	<ul style="list-style-type: none"> Input Start and Stop frequencies LO frequency Output sideband (High or Low) 	<ul style="list-style-type: none"> IF Start and Stop frequencies 2nd LO frequency IF sideband (High or Low)
3 naCalculateLO1	<ul style="list-style-type: none"> Input Start and Stop frequencies Output frequency Output sideband (High or Low) 	<ul style="list-style-type: none"> IF Start and Stop frequencies 2nd LO frequency IF sideband (High or Low)

4 naCalculateLO2 NA

- Input Start and stop frequencies
- 1st LO start and stop frequencies
- Output frequency
- IF sideband(High or Low)
- Output sideband(High or Low)

Return Type Not Applicable

Default Not Applicable

Examples `mxr.SegmentCalculate 1,2 'Calculates the output frequencies for segment 1.'`

[See example program](#)

C++ Syntax HRESULT SegmentCalculate(long index, ConverterCalculation param);

Interface IConverter5

Last Modified:

26-Oct-2010 New command (A.09.33)

SelectCalSet Method

Description Selects and applies a Cal Set to the specified channel.

Note: Error Correction is not automatically applied as a result of this command being issued. If there is more than one Cal Type in the Cal Set, you must explicitly choose the Cal Type you want to apply. (See [meas.Caltype](#))

VB Syntax *channel.SelectCalSet calSet, restore*

Variable [\(Type\)](#) - Description

channel **(object)** - A [Channel](#) object

calSet **(string)** - Cal Set to make active. Specify the Cal Set by GUID or Name. Use [EnumerateCalSets](#) to list the available Cal Sets.

restore **(boolean)** -

True (1) - The stimulus stored with the cal set will be applied to the channel.

False (0) - If a conflict is detected between the existing channel settings and the Cal Set stimulus settings, then the following will occur:

If interpolation is ON, then interpolation will be attempted. This may fail if the channel frequency is outside the range of the Cal Set.

If interpolation is OFF, the selection will be abandoned and an error is returned: E_NA_CAL_STIMULUS_VALUES_EXCEEDED

Return Type Not Applicable

Default Not Applicable

Example `channel.SelectCalSet GUID, 1`

`chan.SelectCalSet "MyCalSet", 0`

C++ Syntax HRESULT SelectCalSet (BSTR strCset, bool bRestore);

Interface IChannel

Last Modified:

29-Nov-2007 Modified to accept name

SetAllSegments Method

Description Uploads a segment table to the PNA replacing any existing segment table.

Segments must be ascending in frequency and non-overlapping. If they are not, the segments are 'adjusted' as they are from the User Interface control. The total number of points for all segments cannot exceed the PNA [maximum number of points](#) for a sweep.

VB Syntax *Segs.SetAllSegments (segdata)*

Variable [\(Type\)](#) - Description

segs A [Segments](#) (Collection)

segdata (Variant) A 2-dimensional array of Segment data:

- dimension 0 is the number of elements in each segment.
- dimension 1 is the number of segments that will be used.

The following is a list of dimension 0 elements for each segment:

Note: All elements must be **dimensioned** as either ALL Double or ALL Variant.

- 0 = Segment state (Boolean True or False)
- 1 = Number of Points in this segment (Integer)
- 2 = Start Freq (Double)
- 3 = Stop Freq (Double)
- 4 = IFBW (Double) optional
- 5 = Dwell Time (Double) optional
- 6 + = Power (Double) optional; see table below.

The first four data elements must always be supplied. After those values, data must be supplied for successive optional elements. For example, to set dwell time values, you must also supply IFBW values, because IFBW (#4) precedes dwell time (#5) in the array order.

The [IFBandwidthOption](#), [SweepTimeOption](#), and [SourePowerOption](#) settings do NOT affect the order in which elements are interpreted.

The number of elements to supply for Power depends on the following two settings:

1. [SourcePowerOption](#) = True allows segments to have independent power levels.
2. [CouplePorts](#) = False allows different power levels for each test port.

CouplePorts	SourcePowerOption	Number of Elements
False	False	Each port has its own channel-wide power setting, which is set using TestPortPower . Provide exactly 7 elements per segment. The last element (power) is ignored.
False	True	Provide 6 elements + total number of ports. The first 7 elements are still interpreted the same. The remaining elements (in-order) are interpreted as the power levels to set on that segment for Ports 2 through N, where N is the total number of ports currently enabled for the PNA or for a PNA with multiport external test set.
True	False	Provide exactly 7 elements per segment. The last element (power) is ignored.
True	True	Provide exactly 7 elements per segment. The last element (power) is honored.

Return Type Not Applicable

Default Not Applicable

Examples [See a VB example using this command](#)

[See a C++ example using this command](#)

C++ Syntax HRESULT SetAllSegments (VARIANT Segments);

Interface ISegments2

Last Modified:

28-Apr-2009 [More edits](#)

15-Oct-2007 [Major edits and link to C++ example](#)

SetBBPorts Method

Description For a Balanced - Balanced device type, maps the PNA ports to the DUT ports.

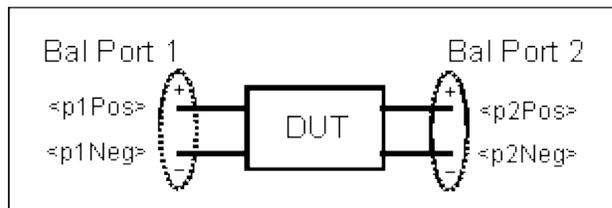
Set the Balanced device type using the [DUTTopology Property](#)

VB Syntax *balTopology*.**SetBBPorts** *p1Pos, p1Neg, p2Pos, p2Neg*

Variable [\(Type\)](#) - Description

balTopology A [BalancedTopology](#) (object)

p1Pos, p1Neg, p2Pos, p2Neg (**Long Integer**) PNA port number that connects to each of the following DUT ports:



Return Type Not applicable - To read port mappings, use the [BalancedTopology](#) properties.

Default Not Applicable

Examples `balTop.SetBBPorts 1,2,3,4`

C++ Syntax HRESULT SetBBPorts (long p1Pos, long p1Neg, long p2Pos, long p2Neg)

Interface IBalancedTopology

SetBSPorts Method

Description For a Balanced-Single-ended device type, maps the PNA ports to the DUT ports.

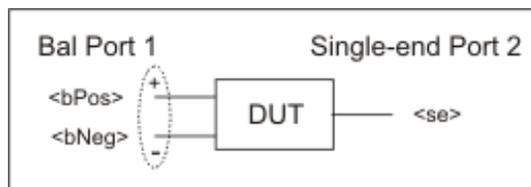
Set the Balanced-Single-ended device type using the [DUTTopology Property](#)

VB Syntax *balTopology.SetBSPorts bPos, bNeg, se*

Variable [\(Type\)](#) - **Description**

balTopology A [BalancedTopology](#) (**object**)

bPos, bNeg, se PNA port number that connects to each of the following DUT ports:



Return Type Not applicable - To read port mappings, use the [BalancedTopology](#) properties.

Default Not Applicable

Examples `balTop.SetBSPorts 1,2,3`

C++ Syntax HRESULT SetBSPorts (long bPos, long bNeg, long se)

Interface IBalancedTopology2

Last modified:

25-Apr-2012 New topic

SetCalInfo Method

Description Specifies the type of Unguided calibration. This method should be the first method called on the calibrator object. It prepares the internal state for the rest of the calibration.

Note: You can NOT perform a 3 or 4-port cal using SetCalInfo even though there is enumCalTypes. You must use the [GuidedCalibration](#) object.

Learn more about [reading and writing Cal data using COM](#)

The analyzer can measure both ports simultaneously, assuming you have two of each standard type. For a 2-port cal, See [cal.Simultaneous2PortAcquisition](#)

VB Syntax *cal*.**SetCalInfo** (*type,rcvPort,srcPort*)

Variable [\(Type\) - Description](#)

cal A Calibrator (**object**)

type (**enum NACalType**) - Calibration type. Choose from:

- 0 - naCalType_Response_Open
- 1 - naCalType_Response_Short
- 2 - naCalType_Response_Thru
- 3 - naCalType_Response_Thru_And_Isol
- 4 - naCalType_OnePort
- 5 - naCalType_TwoPort_SOLT
- 6 - naCalType_TwoPort_TRL
- 7 - naCalType_None
- 8 - naCalType_ThreePort_SOLT
- 9 - Custom
- 10 - naCalType_FourPort_SOLT

Note: For 1-port cals, the source port = receiver port. For 2, 3,4-port SOLT and TRL, it doesn't matter which port is specified as source and receiver

rcvPort (**long integer**) - Receiver Port

srcPort (**long integer**) - Source Port

Return Type NACalType

Default 7- naCalType_None

Examples `cal.setCalInfo(naCalType_Response_Open,1,1)`

C++ Syntax HRESULT SetCalInfo(tagNACalType calType,long portA, long portB)

Interface ICalibrator

SetCalInfoEx Method (for source power cal)

Description	This command replaces SetCalInfo2 Method . Specifies the channel and the source port to be used for the source power calibration about to be performed.
VB Syntax	<code>powerCalibrator.SetCalInfoEx channel, srcPort, [powerOffset,] [display]</code>
Variable	(Type) - Description
<i>powerCalibrator</i>	(object) - A SourcePowerCalibrator object
<i>channel</i>	(long integer) - Number of the PNA channel (not power meter channel) on which the source power cal will be performed. If the channel does not already exist, it will be created.
<i>srcPort</i>	(long integer) - Port number on which the source power cal will be performed. Note: If the port is defined by a string name, such as an external source, a balanced port, or one of the Source 2 outputs on the 2-port 2-source PNA-x model, then you must use chan.getPortNumber to translate the string into a port number. To learn more see Remotely Specifying a Source Port .
<i>[powerOffset]</i>	(double) - Optional argument. Sets or returns a power level offset from the PNA test port power. This can be a gain or loss value (in dB) to account for components you connect between the source and the reference plane of your measurement. For example, specify 10 dB to account for a 10 dB amplifier at the input of your DUT. Following the calibration, the PNA power readouts are adjusted by this value. This argument performs the same function as chan.SourcePowerCalPowerOffset Property
<i>[display]</i>	(boolean) Optional argument. Enables and disables the display of power readings on the PNA screen. After the source power cal data is acquired, this setting is reset to ON. If unspecified, value is set to ON. True - Display of power readings is ON False - Display of power readings is OFF
Return Type	None
Default	Not Applicable
Examples	<code>powerCalibrator.SetCalInfoEx 1, 1, -10, True</code>

C++ Syntax HRESULT SetCalInfoEx(long Channel, long SourcePort, double PowerOffset = 0., VARIANT_BOOL bDisplay = VARIANT_TRUE);

Interface ISourcePowerCalibrator4

Last Modified:

24-Apr-2008 Added note for string names
30-Apr-2007 Edited for src strings

SetCenter Method

Description	Changes the center stimulus to the stimulus value of the marker. The start stimulus stays the same and the stop is adjusted. This command does not work with channels that are in CW or Segment Sweep mode.
VB Syntax	<i>mark</i> .SetCenter
Variable	(Type) - Description
<i>mark</i>	A Marker (object)
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>mark.SetCenter</code>
C++ Syntax	HRESULT SetCenter()
Interface	IMarker

SetCW Method

Description	Changes the analyzer to sweep type CW mode and sets the CW frequency to the marker's frequency. Does not change anything if current sweep type is other than a frequency sweep.
VB Syntax	<i>mark</i> .SetCW
Variable	(Type) - Description
	<i>mark</i> A Marker (object)
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>mark.SetCW</code>
C++ Syntax	HRESULT SetCW()
Interface	IMarker

SetCWFreq Method

Description Sets the CW frequency to the frequency of the active marker. Does NOT change sweep type.

Use ONLY when the current sweep type is sweeping frequency - NOT available in CW or Power Sweep.

Use this command to first set the CW Frequency to a value that is known to be within the current calibrated range, THEN set [Sweep Type](#) to naPowerSweep or naCWTimeSweep.

VB Syntax *mark*.SetCWFreq

Variable [\(Type\)](#) - Description

mark A Marker (**object**)

Return Type Not Applicable

Default Not Applicable

Examples `mark.SetCWFreq`

C++ Syntax HRESULT SetCWFreq()

Interface IMarker3

Last Modified:

3-Oct-2008 MX New topic

SetDutPorts Method

Description	Sets the PNA to DUT port map for FCA measurements. Use DeviceInputPort and DeviceOutputPort to read these values. Changing the ports may limit your ability to use an internal second source. If a selected port is shared by one of the sources, then that source will not be available as an LO source. Learn more about Internal second sources.
VB Syntax	<i>mixer</i> .SetDUTPorts (<i>inputPort</i> , <i>outputPort</i>)
Variable	(Type) - Description
<i>mixer</i>	A IMixer Interface pointer to the Meas (object)
<i>inputPort</i>	(Long) PNA port to be connected to the DUT input. <ul style="list-style-type: none">• For SMC, choose any unused PNA port.• For VMC, set to 1
<i>outputPort</i>	(Long) PNA port to be connected to the DUT output. Choose any unused port for SMC and VMC.
Return Type	Not Applicable
Default	1,2
Examples	<pre>mixer.SetDUTPorts =2,1</pre>
C++ Syntax	HRESULT SetDutPorts(long inputPort,long OutputPort);
Interface	IMixer8

Last Modified:

23-Apr-2008 New topic.

SetElectricalDelay Method

Description Changes the measurement's electrical delay to the marker's delay value.

VB Syntax `mark.SetElectricalDelay`

Variable [\(Type\)](#) - Description

mark A Marker (**object**)

Return Type Not Applicable

Default Not Applicable

Examples `mark.SetElectricalDelay`

C++ Syntax HRESULT SetElectricalDelay()

Interface IMarker

SetFailOnOverRange Method

Description When set TRUE, configures the analyzer to report outOfRange conditions with an **error** code. Any overrange error will return **E_NA_LIMIT_OUTOFRANGE_ERROR**.

Note: This method is for the benefit of VB clients. The analyzer automatically adjusts overrange conditions to the closest acceptable setting. The VB user will not see that an overrange occurred because the HRESULT is not returned if it has a success code. For more information, see [Events/OverRange](#).

VB Syntax `app.SetFailOnOverRange state`

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

state (boolean) -

True (1) - Overrange conditions report an error code

False (0) - Overrange conditions report a success code

Return Type Not Applicable

Default False (0)

VB Example

```
app.SetFailOnOverRange TRUE
On Error Goto ERRHANDLER
```

```
'the following overrange will cause ERRHANDLER to be invoked
```

```
channel.StartFrequency = 9.9 GHZ
exit
```

```
ERRHANDLER:
    print "something failed"
```

C++ Syntax HRESULT put_SetFailOnOverRange(VARIANT_BOOL mode)

Interface IApplication

Write only

SetIsolationPaths Method

Description	Adjusts the list of paths (port pairings) for which isolation standards will be measured during calibration.
VB Syntax	<i>guidedCal</i> . SetIsolationPaths <i>specifier</i> , <i>pathList</i>
Variable	(Type) - Description
<i>obj</i>	Any of the following: GuidedCalibration (object)
<i>specifier</i>	(Enum) - Choose from: 0 - naPathsAll - Measure isolation on all pairings of the ports that are to be calibrated. 1 - naPathsNone - Do not measure isolation on any pairing of the ports to be calibrated. 2 - naPathsAdd - Add one or more specific pairings of ports to the list of port pairings for which isolation will be measured. 3 - naPathsRemove - Remove one or more specific pairings of ports from the list of port pairings for which isolation will be measured.
<i>pathlist</i>	(Variant) port numbers in pairs. One-dimensional array of Long Integers. Note: <i>pathList</i> is evaluated only when <i>specifier</i> is naPathsAdd or naPathsRemove . For naPathsAll and naPathsNone , <i>pathList</i> is ignored.
Return Type	Not Applicable
Default	Not Applicable
Examples	<pre>Dim pathList 'selecting to measure isolation on all possible paths for the ports about to be calibrated guidedCal.SetIsolationPaths naPathsAll, pathList 'now removing the paths 1-to-2, 2-to-3 and 2-to-4 from the set of all paths pathList = Array(1,2,2,3,2,4) guidedCal.SetIsolationPaths naPathsRemove, pathList</pre>
C++ Syntax	HRESULT SetIsolationPaths(enum NAPortPathSpecifier specifier, VARIANT pathList);

Interface IGuidedCalibration3

Last Modified:

16-Apr-2007 MX New topic

SetIPConfiguration Method

Description Modifies settings of the PNA computer networking configuration.

VB Syntax *app.SetIPConfiguration AutoIPAddress, DNSServer1, DNSServer2, HostName, DomainName, IPAddress, SubNet, Gateway, DNSSuffix1, DNSSuffix2*

or

retStr = app.SetIPConfiguration (AutoIPAddress, DNSServer1, DNSServer2, HostName, DomainName, IPAddress, SubNet, Gateway, DNSSuffix1, DNSSuffix2)

Variable [\(Type\) - Description](#)

app An [Application](#) (object)

AutoIPAddress **(boolean)** -Choose either:

True - PNA is assigned an IP address by a DHCP server, or will use AutoIP (Dynamic Link-Local Addressing) if DHCP server not found.

False – PNA will use the static IP address value specified by IPAddress.

DNSServer1 **(string)** IP address of primary DNS server.

When AutoIPAddress = True and an empty string is specified for DNSServer1, the PNA will attempt to obtain the addresses of primary and secondary DNS servers automatically.

When AutoIPAddress = False, an IP address must be specified for DNSServer1 and/or DNSServer2 or else the PNA's host name will not be resolvable on the computer network.

DNSServer2 **(string)** IP address of secondary DNS server. When specifying an empty string for DNSServer1, then specify an empty string here also.

HostName **(string)** DNS host name (computer name) to be assigned to this PNA.

Note: If specifying a name different than the PNA's current host name, the change will not take effect until after you reboot the PNA.

DomainName **(string)** DNS domain name associated with this PNA.

IPAddress **(string)** Static IP address to assign to this PNA when AutoIPAddress = False. When AutoIPAddress = True, the value of IPAddress is ignored.

- SubNet* **(string)** Subnet mask value to assign to the PNA network configuration.
- Gateway* **(string)** Gateway address to assign to the PNA network configuration.
- DNSSuffix1* **(string)** Primary suffix to set in the PNA DNS suffix search order. An empty string is allowed.
- DNSSuffix2* **(string)** Secondary suffix to set in the PNA DNS suffix search order. An empty string is allowed.
- retStr* **(string)** String returned by this method should be ignored. It is intended for Agilent diagnostic use.

Return Type String

Default Not Applicable

Examples

```
app.SetIPConfiguration True, "", "", "MyHostName",  
"MyRegion.MyCompany.com", "", "255.255.255.0", "123.45.67.890",  
"", ""
```

```
app.SetIPConfiguration False, "123.456.78.90", "234.56.78.901",  
"MyHostName", "MyRegion.MyCompany.com", "123.456.789.0",  
"255.255.255.0", "123.45.67.890", "MyCompany.com", ""
```

C++ Syntax HRESULT SetIPConfiguration(VARIANT_BOOL AutoIPAddress, BSTR DNSServer1, BSTR DNSServer2, BSTR HostName, BSTR DomainName, BSTR IPAddress, BSTR SubNet, BSTR Gateway, BSTR DNSSuffix1, BSTR DNSSuffix2, BSTR *pRetStr);

Interface IApplication14

Last Modified:

2-Jun-2008 MX New topic

SetPowerAcquisitionDevice Method

Description Sets the power sensor channel (A or B) to be used. This performs the same function as the **Use this sensor only** checkbox in the Power Sensor Settings dialog.

Note: This method is only necessary when performing an SMC calibration.

VB Syntax *pwrCal* . **SetPowerAcquisitionDevice** *sensor*

Variable (Type) - Description

pwrCal (Object) A SourcePowerCalibrator object

sensor (**enum NAPowerAcquisitionDevice**) The power sensor channel.
Choose from:

0 – naPowerSensor_A

1 – naPowerSensor_B

Default Not Applicable

Examples `pwrCal.SetPowerAcquisitionDevice naPowerSensor_A`

C++ Syntax HRESULT SetPowerAcquisitionDevice(
tagNAPowerAcquisitionDevice enumAcqDevice);

Interface ISourcePowerCalibrator3

Last Modified:

8-May-2012 [Edit example](#)

SetFrequencyLowPass Method

Description Set the start frequencies when **trans.Mode = LowPass**.

VB Syntax *trans*.SetFrequencyLowPass

Variable [\(Type\)](#) - Description

trans A Transform (**object**)

Return Type Not Applicable

Default Not Applicable

Examples `trans.SetFrequencyLowPass`

C++ Syntax HRESULT SetFrequencyLowPass(void)

Interface ITransform

SetPortMap Method

Description Set the DUT-to-PNA port mapping for the [Noise Figure](#), [Gain Compression](#), [IMD](#), [IMDx](#), [IMS](#), or [IMSx](#) measurement.

For Noise Figure:

Port mapping is allowed ONLY when [NoiseReceiver](#) is set to **naStandardReceiver**.

When setting IMD and IMS channels:

- When input is 1, output can be 2 or 4.
- When input is 3, output must be 4.
- This setting is necessary only when using the **limited port mapping feature**. [Learn more.](#)

VB Syntax *obj*.**SetPortMap** *in,out*

Variable [\(Type\)](#) - Description

obj A [GainCompression](#) (**object**) or

A [SweptIMD](#) (**object**) or

An [IMDSpectrum](#) (**object**)

A [NoiseFigure](#) (**object**) - [See example program](#)

in PNA port which is connected to the DUT input.

out PNA port which is connected to the DUT output.

Return Type Not Applicable

To read port map, use:

[DeviceInputPort Property](#)

[DeviceOutputPort Property](#)

Default 1,2

Examples `gca.SetPortMap 2,1`

C++ Syntax HRESULT SetPortMap(long input_port,long output_port);

Interface IGainCompression

ISweptIMD

ISpectrum

INoiseFigure6

Last Modified:

30-Apr-2010	Added NF Opt 028
11-Aug-2009	Added IMD and IMSpectrum (9.0)
29-Nov-2007	MX New topic

SetReferenceLevel Method

Description Changes the measurement's reference level to the marker's Y-axis value.

VB Syntax *mark*.SetReferenceLevel

Variable [\(Type\)](#) - Description

mark A Marker (**object**)

Return Type Not Applicable

Default Not Applicable

Examples `mark.SetReferenceLevel`

C++ Syntax HRESULT SetReferenceLevel()

Interface IMarker

SetSBPorts Method

Description For a Single-ended - Balanced device type, maps the PNA ports to the DUT ports.

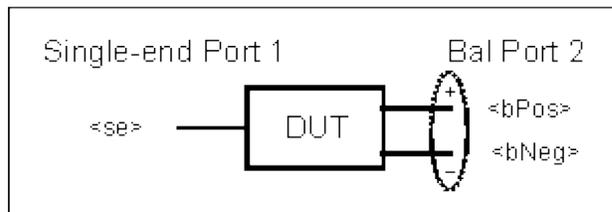
Set the Single-ended - Balanced device type using the [DUTTopology Property](#)

VB Syntax *balTopology.SetSBPorts se, bPos, bNeg*

Variable [\(Type\) - Description](#)

balTopology A [BalancedTopology](#) (object)

se, bPos, bNeg PNA port number that connects to each of the following DUT ports:



Return Type Not applicable - To read port mappings, use the [BalancedTopology](#) properties.

Default Not Applicable

Examples `balTop.SetSBPorts 1,2,3`

C++ Syntax HRESULT SetSBPorts (long se, long bPos, long bNeg)

Interface IBalancedTopology

SetSSBPorts Method

Description For a Single-ended - Single-ended - Balanced device type, maps the PNA ports to the DUT ports.

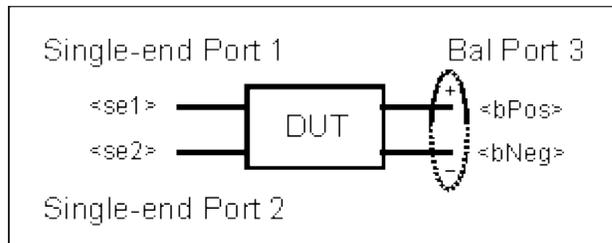
Set the Single-ended - Single-ended - Balanced device type using the [DUTTopology Property](#)

VB Syntax *balTopology.SetSSBPorts se, se2, bPos, bNeg*

Variable [\(Type\)](#) - Description

balTopology A [BalancedTopology](#) (object)

se, se2, bPos, bNeg PNA port number that connects to each of the following DUT ports:



Return Type Not applicable - To read port mappings, use the [BalancedTopology](#) properties.

Default Not Applicable

Examples `balTop.SetSSBPorts 1,2,3,4`

C++ Syntax HRESULT SetSSBPorts (long se, long se2, long bPos, long bNeg)

Interface IBalancedTopology

SetupMeasurementsForStep Method

Description Show the Cal Window, and optionally one or more other specific windows, before acquiring a Cal standard. This command will cause the Cal Window to display the specific measurements that are to be made for that particular Cal standard.

[See custom Cal window commands.](#)

VB Syntax *guidedCal*.SetupMeasurementsForStep (*n*)

Variable [\(Type\)](#) - Description

guidedCal A [GuidedCalibration](#) (object)

n Step number in the calibration process.

Use [GenerateSteps](#) to determine the total number of steps.

Use [GetStepDescription](#) to read the description of each step.

Return Type Not Applicable

Default Not Applicable

Examples `guidedCal.SetupMeasurementsForStep 3`

[See example using this command](#)

C++ Syntax HRESULT SetupMeasurementsForStep(long step);

Interface IGuidedCalibration4

Last Modified:

8-Nov-2007 MX New topic

Write-only

SetStandardsForClass Method

Description Set the calibration standard numbers for a specified calibration class. To read the cal standard numbers use [GetStandardsForClass Method](#)

VB Syntax *calKit.SetStandardsForClass* (*calclassorder, std1, std2, std3, std4, std5, std6, std7*)

Variable [\(Type\)](#) - Description

calKit A CalKit (**object**)

calclassorder (**enum NA CalClassOrder**) Cal Class. Choose from:

- 0 - naRefl_1_S11
- 1 - naRefl_2_S11
- 2 - naRefl_3_S11
- 3 - naTran_1_S21
- 4 - naRefl_1_S22
- 5 - naRefl_2_S22
- 6 - naRefl_3_S22
- 7 - naTran_1_S12
- 8 - naRefl_1_S33
- 9 - naRefl_2_S33
- 10 - naRefl_3_S33
- 11 - naTran_1_S32
- 12 - naTran_1_S23
- 13 - naTran_1_S31
- 14 - naTran_1_S13
- 15 - naTRL_T
- 16 - naTRL_R
- 17 - naTRL_L

std1...std7 (**long**) Calibration Standard Number. Choose from **1** through **30**. Std2 through Std7 are optional

Return Type Not applicable

Default Not applicable

Examples `calkit.SetStandardsForClass naRefl_3_S11, 3, 5, 6`

`calkit.SetStandardsForClass naTran_1_S21, 4`

C++ Syntax `HRESULT SetStandardsForClass(NACalClassOrder calclassorder, long std1, long std2, long std3, long std4, long std5, long std6, long std7)`

Interface `ICalKit`

SetStart Method

Description Changes the start stimulus to the stimulus value of the marker. The stop stimulus stays the same and the span is adjusted.

This command does not work with channels that are in [CW](#) or [Segment Sweep](#) mode.

VB Syntax `mark.SetStart`

Variable [\(Type\) - Description](#)

mark A Marker (**object**)

Return Type Not Applicable

Default Not Applicable

Examples `mark.SetStart`

C++ Syntax HRESULT SetStart()

Interface IMarker

SetStop Method

Description Changes the stop stimulus to the stimulus value of the marker. The start stimulus stays the same and the span is adjusted.

This command does not work with channels that are in [CW](#) or [Segment Sweep](#) mode.

VB Syntax `mark.SetStop`

Variable [\(Type\) - Description](#)

mark A Marker (**object**)

Return Type Not Applicable

Default Not Applicable

Examples `mark.SetStop`

C++ Syntax HRESULT SetStop()

Interface IMarker

ShowMarkerReadout Method

Description	Shows and Hides the Marker readout for the active marker in the upper-right corner of the window.
VB Syntax	<i>win</i> . ShowMarkerReadout <i>state</i>
Variable	(Type) - Description
<i>win</i>	A NAWindow (object)
<i>state</i>	(boolean) - True (1) - Show the Marker readout False (0) - Hide the Marker readout
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>win.ShowMarkerReadout True</code>
C++ Syntax	HRESULT ShowMarkerReadout(VARIANT_BOOL bState)
Interface	INAWindow

ShowStatusBar Method

Description	Shows and Hides the Status Bar. The Status Bar is located across the bottom of the display. The following information is shown for the active measurement: <ul style="list-style-type: none">• Channel number• Parameter• Correction On or Off• Remote or Local operation
VB Syntax	<i>app</i> . ShowStatusBar <i>state</i>
Variable	(Type) - Description
<i>app</i>	An Application (object)
<i>state</i>	(boolean) - True (1) - Show the Status Bar False (0) - Hide the Status Bar
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>app.ShowStatusBar True</code>
C++ Syntax	HRESULT ShowStatusBar (VARIANT_BOOL bState)
Interface	IApplication

ShowStimulus Method

Description	Shows and Hides the Stimulus (X-axis) information located at the bottom of the display. The start and stop stimulus values are shown for the active measurement.
VB Syntax	<i>app</i> . ShowStimulus <i>state</i>
Variable	(Type) - Description
<i>app</i>	An Application (object)
<i>state</i>	(boolean) - True (1) - Show the Stimulus information False (0) - Hide the Stimulus information
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>app.ShowStimulus True</code>
C++ Syntax	HRESULT ShowStimulus(VARIANT_BOOL bState)
Interface	IApplication

ShowTable Method

Description	Shows or Hides the specified table for the window's active measurement in the lower part of the window.
VB Syntax	<i>win.ShowTable value</i>
Variable	(Type) - Description
<i>win</i>	A NAWindow (object)
<i>value</i>	(enum naTable) - The table to show or hide. Choose from: 0 - naTable_None 1 - naTable_Marker 2 - naTable_Segment 3 - naTable_Limit
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>win.ShowTable naTable_limit</code>
C++ Syntax	HRESULT ShowTable (tagNATableType table)
Interface	INAWindow

ShowTitleBars Method

Description Shows and Hides the Title Bars. The Title Bars are across the top of the Network Analyzer Window and each of the measurement windows. The Window name is shown in the Title Bar.

VB Syntax *app.ShowTitleBars state*

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

state (boolean)
True (1) - Show the Title Bars
False (0) - Hide the Title Bars

Return Type Not Applicable

Default Not Applicable

Examples `app.ShowTitleBars True`

C++ Syntax HRESULT ShowTitleBars/(VARIANT_BOOL bState)

Interface IApplication

ShowToolbar Method

Description	Shows and Hides the specified Toolbar.
VB Syntax	<i>app</i> . ShowToolbar <i>toolbar, state</i>
Variable	(Type) - Description
<i>app</i>	An Application (object)
<i>toolbar</i>	(enum NAToolbarType) - The toolbar to show or hide. Choose from: 0 - naToolbar_None 1 - naToolbar_ActiveEntry 2 - naToolbar_Markers 3 - naToolbar_Measurement - OBSOLETE 4 - naToolbar_Stimulus - OBSOLETE 5 - naToolbar_SweepControl - OBSOLETE 6 - naToolbar_Transform 7 - naToolbar_PortExtensions 8 - naToolbar_Keys
<i>state</i>	(boolean) - True (1) - Show the specified toolbar False (0) - Hide the specified toolbar
Return Type	Not Applicable
Default	1 - naToolbar_ActiveEntry showing; all others hiding.
Examples	<code>app.ShowToolbar 1,1 'shows the active entry toolbar</code>
C++ Syntax	HRESULT ShowToolbar(tagNAToolbarType toolbar, VARIANT_BOOL bState)
Interface	IApplication

Last Modified:

21-Jun-2013 Removed three toolbars (10.0) - added three more

Single Method

Description Sets the trigger count to 1 which will cause the channel to respond once to the [trigger source](#).

How the channel responds to a single trigger depends on the [trigger mode](#) (point, trace, and so forth.)

With the exception of the 'sync' argument, this command behaves like the [channel 'single' setting](#) from the user interface.

This setting has implications on Calibration. [Learn more](#).

VB Syntax `chan.Single [sync]`

Variable **(Type) - Description**

`chan` A Channel (**object**)

`[sync]` (**boolean**) -Optional argument.

- **True** - The PNA waits (blocks execution) until the entire acquisition process is completed.
- **False** - The PNA returns immediately - does NOT wait for acquisition to complete (non-blocking). Default setting.

When trigger source is set to Manual:

- with `sync = True`, trigger source automatically changes to Internal which sends AND allows one trigger signal, then changes back to Manual.
- with `sync = False`, a trigger signal must also be sent using [app.ManualTrigger Method](#).

Return Type Not Applicable

Default Not Applicable

Examples

```
sync = True
chan.Single sync
```

C++ Syntax HRESULT Single(VARIANT_BOOL bWait)

Interface IChannel

10-Jun-2009 Minor edits
10-Jul-2007 Corrected for manual mode with sync = True

Store Method

Description Saves the path configuration currently associated with channel (ch) to the specified configuration name.

This command is identical to PathConfigurationManager.[StoreConfiguration Method](#)

VB Syntax *pathMgr.StoreConfiguration ch, name*

Variable [\(Type\) - Description](#)

pathMgr [PathConfigurationManager \(object\)](#)

ch **(Long)** Channel number of the configuration to be saved.

name **(String)** Configuration name. Factory configurations can NOT be overwritten. Specifying the name of a pre-defined factory configuration will result in an error.

Return Type Not Applicable

Default Not Applicable

Examples `path.StoreConfiguration(2) "myMixer"`

C++ Syntax HRESULT StoreConfiguration(long channelNum, BSTR configName);

Interface IPathConfigurationManager

Last modified:

Dec.12, 2006 MX New Command

StoreConfiguration Method

Description	Saves the path configuration currently associated with channel (ch) to the specified configuration name.
VB Syntax	<i>pathMgr.StoreConfiguration ch, name</i>
Variable	(Type) - Description
<i>pathMgr</i>	PathConfigurationManager (object)
<i>ch</i>	(Long) Channel number of the configuration to be saved.
<i>name</i>	(String) Configuration name. Factory configurations can NOT be overwritten. Specifying the name of a pre-defined factory configuration will result in an error.
Return Type	Not Applicable
Default	Not Applicable
Examples	<pre>path.StoreConfiguration(2) "myMixer"</pre>
C++ Syntax	HRESULT StoreConfiguration(long channelNum, BSTR configName);
Interface	IPathConfigurationManager

Last Modified:

14-Dec-2006 MX New topic

StoreTheme Method

Description Saves the current color theme to a disc file.

VB Syntax *colors*.**StoreTheme** (*filename*)

Variable [\(Type\)](#) - Description

colors A [ComColors](#) (**object**)

filename (**String**) - Path and filename of the theme to save.

Return Type Not Applicable

Default Not Applicable

Examples

```
colors.StoreTheme = ("c:/Program Files/Agilent/Network Analyzer/Colors/Theme1.colors")
```

C++ Syntax HRESULT StoreTheme(BSTR filename);

Interface IComColors

StringToNACalClass Method

Description Converts the returned strings from [GetStandardsList](#) into the enumeration (NACalClass) and the port numbers required for [PutStandard](#) and [GetStandard](#) methods that transmit data in and out of the Cal Set.

Learn more about [reading and writing Cal data using COM](#)

VB Syntax `CalSet.StringToNACalClass (list, std, rcv, src)`

Variable [\(Type\) - Description](#)

CalSet **(object)** - A Cal Set object

list **(string)** - a string containing the textual description of the standard.

std **(enum NACalClass)** Choose from:

- 1 - naClassA
- 2 - naClassB
- 3 - naClassC
- 4 - naClassD
- 5 - naClassE
- 6 - naReferenceRatioLine
- 7 - naReferenceRatioThru

SOLT Standards

- 1 - naSOLT_Open
- 2 - naSOLT_Short
- 3 - naSOLT_Load
- 4 - naSOLT_Thru
- 5 - naSOLT_Isolation

TRL Standards

- 1 - naTRL_Reflection
- 2 - naTRL_Line_Reflection
- 3 - naTRL_Line_Tracking
- 4 - naTRL_Thru
- 5 - naTRL_Isolation

rcv **(long)** - port number of the receiver

src (**long**) - port number of the source

Return Type Not Applicable

Default Not Applicable

Examples `guid = CalSet.StringToNACalClass(list, std, rcv, src)`

C++ Syntax HRESULT StringtoNACalClass (BSTR* str, NACalClass* item, long *rcv, long *src);

Interface ICalSet

StringToNAErrorTerm2 Method

Description Converts the returned strings from [GetErrorTermList](#) into the enumeration (NAErrorTerm2) and the port numbers required for [PutErrorTerm](#) and [GetErrorTerm](#) methods that transmit data in and out of the Cal Set.

Learn more about [reading and writing Cal data using COM](#)

VB Syntax *Cal Set*.StringToNAErrorTerm2 (*list*, *eterm*, *rcv*, *src*)

Variable [\(Type\) - Description](#)

Cal Set **(object)** - A Cal Set object

list **(string)** - a string containing the textual description of the error term.

eterm **(enum As NaErrorTerm2)**. Choose from:

0 - naET_Directivity (rcv = src)

1 - naET_SourceMatch (rcv = src)

2 - naET_ReflectionTracking (rcv = src)

3 - naET_TransmissionTracking (rcv != src)

4 - naET_LoadMatch (rcv != src)

5 - naET_Isolation (rcv != src)

rcv **(long)** - port number of the receiver

src **(long)** - port number of the source

Return Type Not Applicable

Default Not Applicable

Examples `CalSet.StringToNAErrorTerm2 str, term, rcv, src`

C++ Syntax HRESULT StringToNAErrorTerm2 (BSTR* str, NAErrorTerm2* item, long *rcv, long *src);

Interface ICalSet

SweepOnlyCalChannelDuringCalAcquisition Method

Description	Clears ALL flags for channels to sweep during calibration except the Cal channel. To flag a channel, see AllowChannelToSweepDuringCalAcquisition Method
VB Syntax	<i>calMgr</i> .SweepOnlyCalChannelDuringCalAcquisition
Variable	(Type) - Description <i>calMgr</i> (object) - A CalManager object
Return Type	Not Applicable
Default	Not Applicable
Example	<pre>calMgr.SweepOnlyCalChannelDuringCalAcquisition</pre> See example using this command
C++ Syntax	HRESULT SweepOnlyCalChannelDuringCalAcquisition()
Interface	ICalManager5

Last Modified:

8-Nov-2007 MX New topic

TestsetCatalog Method

Description Returns a list of supported testsets.

VB Syntax *data* = *Tsets*.**TestsetCatalog**

Variable [\(Type\)](#) - Description

data (**variant array**) - Variable to store the returned data.

Tsets (**object**) - An [ExternalTestSets](#) collection

Return Type Variant

Default Not Applicable

Examples `value = Tsets.TestsetCatalog`

C++ Syntax HRESULT TestsetCatalog (VARIANT* Data);

Interface IExternalTestSets

UserPreset Method

Description	Performs a User Preset. There must be an active User Preset state file (see UserPresetLoadFile and UserPresetSaveState) or an error will be returned. Regardless of the state of the User Preset Enable checkbox, the app.Preset command will always preset the PNA to the factory preset settings, and app.UserPreset will always perform a User Preset.
VB Syntax	<i>app</i> . UserPreset
Variable	(Type) - Description
	<i>app</i> An Application (object)
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>app.UserPreset</code>
C++ Syntax	HRESULT UserPreset()
Interface	IApplication6

UserPresetLoadFile Method

Description Loads an existing instrument state file (.sta or .cst) to be used for User Preset. Subsequent execution of [app.UserPreset](#) will cause the PNA to assume this instrument state.

Regardless of the state of the [User Preset Enable](#) checkbox, the [app.Preset](#) command will always preset the PNA to the factory preset settings, and [app.UserPreset](#) will always perform a User Preset.

VB Syntax `app.UserPresetLoadFile (file)`

Variable [\(Type\)](#) - Description

app An [Application](#) (**object**)

file (**String**) Full path, name, and extension of the file to be loaded.

Return Type Not Applicable

Default Not Applicable

Examples `app.UserPresetLoadFile ("C:/Program Files/Agilent/Network Analyzer/Documents/10MHzto20GHz.sta")`

C++ Syntax HRESULT UserPresetLoadFile (BSTR bstrFile)

Interface IApplication6

UserPresetSaveState Method

Description Saves the current instrument settings as UserPreset.sta. Subsequent execution of [app.UserPreset](#) will cause the PNA to assume this instrument state.

Regardless of the state of the [User Preset Enable](#) checkbox, the [app.Preset](#) command will always preset the PNA to the factory preset settings, and [app.UserPreset](#) will always perform a User Preset.

VB Syntax `app.UserPresetSaveState`

Variable [\(Type\)](#) - Description

app An [Application](#) (object)

Return Type Not Applicable

Default Not Applicable

Examples `app.UserPresetSaveState`

C++ Syntax HRESULT UserPresetSaveState()

Interface IApplication6

WriteData Method

Description	Writes a 13-bit value to the specified address using the AD0 through AD12 lines of the external test set connector. The PNA generates the appropriate timing signals. It automatically controls timing signals LDS, LAS and RLW to strobe the address, then the data, to the external test set. See the timing diagram for Address and Data I/O read.
VB Syntax	<i>ExtIO</i> . WriteData <i>address, value</i>
Variable	(Type) - Description
<i>ExtIO</i>	(object) - An External IO object
<i>address</i>	(variant) - Address to be written to.
<i>value</i>	(variant) - 13-bit word to write
Return Type	Not Applicable
Default	Not Applicable
Examples	<code>ExtIO.WriteData 15,12</code>
C++ Syntax	HRESULT WriteData(VARIANT Address, VARIANT Data);
Interface	IHWExternalTestSetIO

Last Modified:

27-Nov-2012 Fixed

WriteRaw Method

Description Writes a 16-bit value to the external test set connector lines AD0 - AD12, RLW, LAS and LDS. The analyzer does NOT generate the appropriate timing signals. The user has control of all 16 lines using this write method.

Note: When RLW (pin25) is set to 1 (high) it causes lines AD0 - AD12 to float. It disables their output latches and sets the hardware for reading. LDS and LAS are not affected by this behavior.

Below is the format of data that is written with WriteRaw:

* This Output will float if RLW (bit-13) is set high

Pin	Bit	Signal name
22	0	AD0*
23	1	AD1*
11	2	AD2*
10	3	AD3*
9	4	AD4*
21	5	AD5*
20	6	AD6*
19	7	AD7*
6	8	AD8*
5	9	AD9*
4	10	AD10*
17	11	AD11*
3	12	AD12*
25	13	RLW
24	14	LDS
8	15	LAS

VB Syntax *ExtIO.WriteRaw value*

Variable [\(Type\)](#) - Description

ExtIO **(object)** - An External IO object

value **(variant)** - Data to be written

Return Type Not Applicable

Default Not Applicable

Examples `ExtIO.WriteRaw 12`

C++ Syntax HRESULT WriteRaw(VARIANT Output);

Interface IHWExternalTestSetIO

WriteSnpFileWithSpecifiedPorts Method

Description **Note:** This command replaces [app.Save \(.snp\)](#). This command is more explicit regarding the data to be saved, and works for PNAs with multiport test sets.

Saves SNP data to the specified file. [Learn more about SNP data.](#)

VB Syntax *data* = *meas*.**WriteSnpFileWithSpecifiedPorts** *ports*, *filename*

Variable **(Type) - Description**

data **(Variant)** array to store the data.

meas A [Measurement \(object\)](#)

ports **(Variant Array)** One dimensional array containing a list of port numbers for which snp data is requested.

filename **(string)** - Full path, filename, and suffix to store the data.

The suffix is not checked for accuracy. If saving 2 ports, specify "filename.s2p"; If saving 3 ports, specify "filename.s3p." and so forth.

SNP data can be output using several data formatting options. See [SNPFormat Property](#)

Return Type Variant array - automatically dimensioned to the size of the data.

Default Not Applicable

Examples

```
'This VBScript example can be pasted into a notepad file and run  
on the PNA as a macro. Learn how.
```

```
Set pna = CreateObject("AgilentPnA835x.application")
```

```
Set meas = pna.ActiveMeasurement
```

```
'List the port numbers for required data
```

```
ports = Array(1,2,4)
```

```
'specify where to save the data
```

```
filename="C:/Program Files/Agilent/Network  
Analyzer/Documents/MyData.s3p"
```

```
meas.WriteSnpFileWithSpecifiedPorts ports,filename
```

C++ Syntax HRESULT WriteSnpFileWithSpecifiedPorts(VARIANT portsToMeasure,BSTR filename);

Last modified:

9/18/06 MQ Added for multiport

OnCalEvent

Description Triggered by a calibration event. See a list of [CAL Events](#).

Note: Some Severe Events are also used as Error Messages

VB Syntax Sub *app*_**OnCalEvent**(ByVal *eventID* As Variant, ByVal *chanNum* As Variant, ByVal *measNum* As Variant)

Variable [\(Type\) - Description](#)

app An Application (**object**)

eventID Code number of the event which occurred

chanNum Channel Number of the event

measNum Measurement Number of the event

Return Type Not Applicable

Default Not Applicable

Examples

```
Sub pna_OnCalEvent(ByVal eventID As Variant, ByVal  
channelNumber As Variant, ByVal measurementNumber As Variant)  
'  
MsgBox ("A Calibration event has occurred")  
End Sub
```

C++ Syntax HRESULT OnCalEvent(VARIANT eventID, VARIANT channelNumber, VARIANT measurementNumber)

Interface IApplication

Selected Cal Events

512 [naEventID_CAL_QUESTIONABLE](#)

513 [naEventID_CAL_STD_NEEDED](#)

514 [naEventID_CAL_STATE_NOT_HW_COMPATIBLE](#)

515 [naEventID_CAL_REQUIRED](#)

516 [naEventID_CAL_CORRECTION_TURNED_OFF](#)

517 [naEventID_CAL_CORRECTION_TURNED_OFF_INTERPOLATION_OFF](#)

518 [naEventID_CAL_CORRECTION_RESTORED](#)

519 [naEventID_CAL_CORRECTION_TURNED_OFF_FREQRANGE_EXCEEDED](#)

520 [naEventID_CAL_CALTYPE_SET_TO_NONE](#)

521 [naEventID_CAL_CORRECTION_TURNED_OFF_NOT_AN_SPARAM](#)

524 [naEventID_SOURCE_POWER_CAL_COMPLETED](#)

592 [naEventID_SOURCE_POWER_CAL_NOT_PRESENT](#)

593 [naEventID_SOURCE_POWER_CAL_NOT_COMPLETE](#)

594 [naEventID_SOURCE_POWER_CAL_REMOVE_TRACE](#)

595 [naEventID_SOURCE_POWER_CAL_REMOVE_MEAS](#)

596 [naEventID_SOURCE_POWER_CAL_POWER_CHANGED](#)

598 [naEventID_INSUFFICIENT_SLIDE_MOVEMENT](#)

613 [naEventID_CALSET_NOT_FOUND](#)

615 [naEventID_CALSET_CREATED](#)

617 [naEventID_CALSET_FILE_NOT_VALID](#)

634 [naEventID_CALSET_LOAD_FAILED](#)

635 [naEventID_CALSET_SAVE_FAILED](#)

636 [naEventID_CALSET_DELETED](#)

637 [naEventID_CALSET_FILE_NOT_COMPATIBLE](#)

639 [naEventID_NEW_CALSET_FILE_CREATED](#)

640 [naEventID_CAL_SET_IN_USE](#)

644 [naEventID_CAL_COULD_NOT_TURN_ON](#)

693 [naEventID_ERROR_FIXTURING_S2PFILE_CANNOT_OPEN](#)

696 [naEventID_ERROR_FIXTURING_TURNED_OFF](#)

701 [naEventID_MORE_THRU_PATHS_NEEDED](#)

See Also

[Errors and the SCPIStringParser Object](#)

Last modified:

Nov. 6, 2006 Added events

OnChannelEvent

Description	Triggered by a channel event.
VB Syntax	Sub <i>app</i> _ OnChannelEvent (ByVal <i>eventID</i> As Variant, ByVal <i>chanNum</i> As Variant)
Variable	(Type) - Description
<i>app</i>	An Application (object)
<i>eventID</i>	Code number of the event which occurred
<i>chanNum</i>	Channel Number of the event
Return Type	Not Applicable
Default	Not Applicable
Examples	<pre>Sub pna_OnChannelEvent(ByVal eventID As Variant, ByVal channelNumber As Variant) If eventID=naEventID_CHANNEL_CREATED then MsgBox "Channel" + chanelNumber + "was created" End If End Sub</pre>
C++ Syntax	HRESULT OnChannelEvent(VARIANT eventID, VARIANT channelNumber)
Interface	IApplication

Selected Channel Events

- 1792 [naEventID_CHANNEL_SWEEP_COMPLETE](#)
- 1793 [naEventID_CHANNEL_TRIGGER_COMPLETE](#)
- 1796 [naEventID_SET_CHANNEL_DIRTY](#)
- 1797 [naEventID_CLEAR_CHANNEL_DIRTY](#)
- 1801 [naEventID_ALL_SWEEPS_COMPLETED_AND_PROCESSED](#)
- 1805 [naEventID_CHANNEL_CREATED](#)
- 1806 [naEventID_CHANNEL_DELETED](#)
- 1876 [naEventID_NO_SOURCE_ATTEN](#)
- 1879 [naEventID_FREQ_OFFSET_OVERRANGE_SO_TURNED_OFF](#)

1883 [naEventID_PORT_NUMBER_OUT_OF_RANGE](#)

See Also

[Errors and the SCPIStringParser Object](#)

Last modified:

March 2, 2007	Added channel create and delete
Nov. 6, 2006	Added events

OnDisplayEvent

Description Triggered by a display event.

VB Syntax Sub *app_OnDisplayEvent*(ByVal *eventID* As Variant, ByVal *winNum* As Variant, ByVal *traceNum* As Variant)

Variable [\(Type\)](#) - Description

app An Application (**object**)

eventID Code number of the event which occurred

winNum Window Number of the event

traceNum Trace Number of the event

Return Type Not Applicable

Default Not Applicable

Examples

```
Sub pna_OnDisplayEvent(ByVal eventID As Variant, ByVal  
windowNumber As Variant, ByVal traceNumber As Variant)  
MsgBox ("A Display event has occurred")  
End Sub
```

C++ Syntax HRESULT OnDisplayEvent(VARIANT eventID, VARIANT windowNumber, VARIANT traceNumber)

Interface IApplication

Selected Display Events

1541 [naEventID_PRINT_SETUP_FAILURE](#)

1542 [naEventID_PRINT_CANCELED](#)

See Also

[Errors and the SCPIStringParser Object](#)

Last modified:

Nov. 6, 2006 Added events

OnHardwareEvent

Description Triggered by a hardware event. See a list of [Hardware Events](#)

Note: Some Severe Events are also used as Error Messages

VB Syntax Sub *app*_**OnHardwareEvent**(ByVal *eventID* As Variant)

Variable [\(Type\)](#) - Description

app An Application (**object**)

eventID Code number of the event which occurred

Return Type Not Applicable

Default Not Applicable

Examples

```
Private Sub pna_OnHardwareEvent(ByVal eventID As Variant)
MsgBox ("A Hardware event has occured")
End Sub
```

C++ Syntax HRESULT OnHardwareEvent(VARIANT eventID)

Interface IApplication

Selected Hardware Events

848 [naEventID_PHASELOCK](#)

852 [naEventID_RFPOWEROFF](#)

853 [naEventID_RFPOWERON](#)

855 [naEventID_UNLEVELED](#)

857 [naEventID_OVERLOAD](#)

914 [naEventID_TRIGGER_REQUIRES_EDGE_LEVEL_TRIGGER](#)

915 [naEventID_TRIGGER_REQUIRES_TRIGGER_OUT](#)

See Also

Errors and the SCPIStringParser Object

Last modified:

Nov. 6, 2006 Added events

OnMeasurementEvent

Description Triggered by a measurement event.

VB Syntax Sub *app_OnMeasurementEvent*(ByVal *eventID* As Variant, ByVal *measNum* As Variant)

Variable [\(Type\) - Description](#)

app An Application (**object**)

eventID Code number of the event which occurred

measNum Measurement Number of the event

Return Type Not Applicable

Default Not Applicable

Examples

```
Private Sub pna_OnMeasurementEvent(ByVal eventID As Variant, ByVal
measurementNumber As Variant)

MsgBox ("A Measurement event has occured")

End Sub
```

C++ Syntax HRESULT OnMeasurementEvent(VARIANT eventID, VARIANT measurementNumber)

Interface IApplication

Selected Measurement Events

1024 [naEventID_NO_VALID_MEMORY_TRACE](#)

1028 [naEventID_LIMIT_FAILED](#)

1029 [naEventID_LIMIT_PASSED](#)

1034 [naEventID_MEMORY_NOT_SAVED](#)

1035 [naEventID_SET_AVERAGE_COMPLETE](#)

1036 [naEventID_CLEAR_AVERAGE_COMPLETE](#)

1111 [naEventID_MARKER_BANDWIDTH_NOT_FOUND](#)

1112 [naEventID_PEAK_NOT_FOUND](#)

1113 [naEventID_TARGET_VALUE_NOT_FOUND](#)

See Also

[Errors and the SCPIStringParser Object](#)

Last modified:

Nov. 6, 2006 Added events

OnSCPIEvent

Description Triggered by a SCPI event.

Note: Some Severe Events are also used as Error Messages

VB Syntax Sub *app*_**OnSCPIEvent**(ByVal *eventID* As Variant)

Variable [\(Type\)](#) - Description

app An Application (**object**)

eventID Code number of the event which occurred

Return Type Not Applicable

Default Not Applicable

Examples

```
Private Sub pna_OnSCPIEvent(ByVal eventID As Variant)
  MsgBox ("A SCPI event has occurred")
End Sub
```

C++ Syntax HRESULT OnSCPIEvent(VARIANT eventID)

Interface IApplication

Selected SCPI Parser Events

1281 [naEventID_NOTHING_TO_SAY](#)

1284 [naEventID_SCPI_STATUS_BYTE_CHANGE](#)

1360 [naEventID_BAD_SCPI_EXECUTE](#)

1375 [naEventID_CALC_MEASUREMENT_SET_TO_NONE](#)

See Also

[Errors and the SCPIStringParser Object](#)

Last modified:

Nov. 6, 2006 Added events

OnSystemEvent

Description Triggered by a system event. See a list of [System Events](#), also known as general events.

See also [EnableSourceUnleveledEvents Property](#)

Note: Some Severe Events are also used as Error Messages

VB Syntax Sub *app*_**OnSystemEvent**(ByVal *eventID* As Variant)

Variable [\(Type\) - Description](#)

app An Application (**object**)

eventID Code number of the event which occurred

chanNum Channel Number of the event

Return Type Not Applicable

Default Not Applicable

Examples

```
Private Sub pna_OnSystemEvent(ByVal eventID As Variant)
MsgBox ("A System event has occurred")
End Sub
```

C++ Syntax HRESULT OnSystemEvent(VARIANT eventID)

Interface IApplication

Selected System Events

2048 [naEventID_OPTION_NOT_INSTALLED](#)

2049 [naEventID_FEATURE_NOT_AVAILABLE](#)

2050 [naEventID_FEATURE_NOT_VALID](#)

2051 [naEventID_SAVEFILE_OK](#)

2063 [naEventID_RECALLFILE_SUCCESS](#)

2130 [naEventID_PRINTER_TROUBLE](#)

2133 [naEventID_TRIGGERDENIED](#)

2134 [naEventID_MACRO_FAILED](#)

2144 [naEventID_NO_LICENSE](#)

2163 [naEventID_PRESET](#)

2166 [naEventID_TRIGGERFAILED](#)

See Also

[Errors and the SCPIStrParser Object](#)

Last modified:

Nov. 6, 2006 Added events

OnUserEvent

Description Reserved for future use.

VB Syntax Sub app_**OnUserEvent**

C++ Example

The following example uses the smart pointer created by Microsoft Visual Studio. The calls to CoInitialize and CoUninitialize open and close the COM libraries.

Also notice that the pointers local to the main routine are explicitly released. When smart pointers go out of scope, they will perform this duty implicitly. However, we are calling CoUninitialize before they have the chance to be destroyed, so we are obliged to release them.

```
// An example program to illustrate the use of #import to bind to the
// PNA type library.
//

#ifdef _UNICODE
#define _UNICODE
#endif

#include "stdafx.h"
#include "stdio.h"
#include "math.h"

////////////////////////////////////
// import the network analyzer type library
////////////////////////////////////
#import "C:/Program Files/Common Files/Agilent/Pna/835x.tlb" no_namespace,
named_guids
////////////////////////////////////
// include the error definitions for the PNA so we can implement
// error handling.
////////////////////////////////////
#include "C:/Program Files/Common Files/Agilent/Pna/errorsystemmessage.h"

IApplicationPtr pNA; // top level application pointer
float fScalarData [1601]; // global buffer for data retrieval
float fScalarData2[1601];

DWORD dwCookie;

////////////////////////////////////
// SetupChannel:
//
// input: pointer to the channel
//
// function: sets properties on the channel
////////////////////////////////////
void SetupChannel(IChannelPtr pChannel)
{
    pChannel->put_StartFrequency( 1.2E9 );
}
```

```

pChannel->put_StopFrequency ( 4.2E9 );
pChannel->put_NumberOfPoints ( 201);

}

////////////////////////////////////
// AcquireData:
//
// input: pointer to the channel
//
// function: single sweeps the channel
////////////////////////////////////
void AcquireData( IChannelPtr pChannel )
{
    pChannel->Single( TRUE );
}

////////////////////////////////////
// ReadData:
//
// input: pointer to the Measurement object
//
// function: reads data from the measurment's formatted
// result data buffer
////////////////////////////////////
void ReadScalarData(IMeasurementPtr pMeas )
{
    IArrayTransferPtr pDataTransfer;
    pDataTransfer = pMeas;
    long numVals = 1601;
    float* pData = fScalarData;

    if(pDataTransfer){

        pDataTransfer->getScalar( naMeasResult, naDataFormat_LogMag, &numVals, pData);

        for (int i = 0; i < numVals; i++)
            printf("%d\t%f/n",i,pData[i]);
        }
    TCHAR msg[100];
    BSTR param;
    pMeas->get_Parameter(&param);
    swprintf(msg,L"Review %s data",param);
    MessageBox(NULL,msg,L"User Message",0);
    ::SysFreeString(param);
}

void ReadComplexData(IMeasurementPtr pMeas )
{
    IArrayTransferPtr pDataTransfer;
    pDataTransfer = pMeas;
}

```

```

long numVals = 1601;
float* pReal= fScalarData;
float* pImag = fScalarData2;

if(pDataTransfer){

pDataTransfer->getPairedData( naRawData, naRealImaginary, &numVals, pReal, pImag);

for (int i = 0; i < numVals; i++)
printf("%d/t%f/t%f/n",i,pReal[i], pImag[i]);
}
TCHAR msg[100];
BSTR param;
pMeas->get_Parameter(&param);
swprintf(msg,L"Review %s data",param);
MessageBox(NULL,msg,L"User Message",0);
::SysFreeString(param);
}
////////////////////////////////////
// PutData:
//
// input: pointer to the Measurement object
//
// function: writes data to the measurement's raw data
// buffer
////////////////////////////////////
void PutData( IMeasurementPtr pMeas )
{
IArrayTransferPtr pDataTransfer;
pDataTransfer = pMeas;
long numVals = 201;

if(pDataTransfer){

NAComplex* pComplex = new NAComplex[numVals];

pComplex[0].Im = 0;
pComplex[0].Re = 1;
for (int i = 1; i < numVals; i++)
{
pComplex[i].Im = (float)sin(i)/i;
pComplex[i].Re = (float)cos(i)/i;
}

pDataTransfer->putNAComplex( naRawData, numVals, pComplex, naDataFormat_Polar);
delete [] pComplex;
}
}

////////////////////////////////////
// printError
////////////////////////////////////

```

```

void printError( HRESULT hr)
{
    BSTR text;

    hr = pNA->get_MessageText ((NAEventID) hr, &text);
    MessageBox(NULL,text,L"Network Analyzer error",0);
    ::SysFreeString(text);
}

////////////////////////////////////
// main
////////////////////////////////////
int main(int argc, char* argv[])
{
    HRESULT hr;
    const long channel1 = 1;
    const long window1 = 1;
    const long srcport = 1;
    IMeasurementPtr pMeasurement;
    IChannelPtr pChannel;

    // initialize COM libraries
    CoInitialize(NULL);

    try {
        pNA = IApplicationPtr("AgilentPNA835x.Application.1");

        pNA->put_Visible(TRUE);
        pNA->Reset();

        pNA->CreateMeasurement (channel1, "S21",srcport, 3);
        hr = pNA->get_ActiveChannel( &pChannel);

        if (SUCCEEDED (hr))
        {
            SetupChannel( pChannel);
            AcquireData(pChannel);
        }

        hr= pNA->get_ActiveMeasurement( &pMeasurement);
        if (SUCCEEDED(hr))
        {
            pMeasurement->put_Format( naDataFormat_Polar);
            ReadScalarData( pMeasurement);
            ReadComplexData( pMeasurement);
            PutData(pMeasurement);
        }
        if (FAILED(hr))
        {
            printError(hr);
        }
    }
}

```

```
}

// make sure to release the remaining pointers
// before calling CoUninitialize

pMeasurement.Release();
pChannel.Release();
pNA.Release();
}
catch (_com_error err)
{
    printError( err.Error() );
}

CoUninitialize();
return 0;
}
```

Reading Cal Set Data using COM

This example iterates over the entire collection of Cal Sets that currently reside in the PNA. It reads the entire list of error term strings from each Cal Set and queries the data for each term. It then does the same for the standards data.

Learn more about [Reading and Writing Calibration data using COM](#).

Learn more about [Cal Sets](#).

See example: [Writing Cal Set Data using COM](#)

See Other COM Example Programs

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save on the PNA hard drive as CalSets.vbs. [Learn how to setup and run the macro](#).

```
Dim pna
Dim cset
Dim calsets

' create the pna object
' to run on a remote PC, substitute 'name' for the full computer name of your PNA
' to run as a macro on the PNA, remove , "name"
Set pna = CreateObject("AgilentPNA835x.Application", "name")
wscript.echo pna.IDString
' obtain the calset collection
Set calsets = pna.GetCalManager.calsets

' loop thru the calsets
Dim c
For c = 1 To calsets.count
Set cset = calsets.Item(c)

' wscript.echo prints values to a message box
wscript.echo "calset = ", cset.GetGUID, cset.Description

' iterate through error terms data
Dim vterms, fdata
vterms = cset.GetErrorTermList2(0, "")
if (Not IsEmpty(vterms)) then
For i = LBound(vterms) To UBound(vterms)
wscript.echo vterms(i)
vdata = cset.GetErrorTermByString(0, vterms(i))
fdata = cset.GetErrorTermStimulus(0, vterms(i))
```

```
wscript.echo vdata(1,0), vdata(1,1)
wscript.echo fdata(1,0), fdata(1,1)
Next
end if

'   iterate through standards data
vterms = cset.GetStandardList2("")
if (Not IsEmpty(vterms)) then
For i = LBound(vterms) To UBound(vterms)
wscript.echo vterms(i)
vdata = cset.GetStandardByString( vterms(i) )
wscript.echo vdata(1,0), vdata(1,1)
Next
end if
Next
```

Last modified:

2-May-2011 Added GetErrorTermStimulus

Getting Trace Data from the Analyzer

This Visual Basic program:

- Retrieves Scalar Data from the Analyzer and plots it.
- Retrieves Paired Data from the Analyzer and plots it.
- Retrieves Complex Data from the Analyzer and plots it.

To use this code, prepare a form with the following:

- Two MSCharts named **MSChart1** and **MSChart2**
- Three buttons named **GetScalar**, **GetPaired**, **GetComplex**

Note: You can get MSChart in Visual Basic by clicking **Project / Components / Microsoft Chart Control**

```
'Put this in a module
Public dlocation As NADataStore
Public numpts As Long
Public fmt As NADataFormat
Public app As Application
Public measData As IArrayTransfer
Public chan As Channel

Sub Form_Load()
'Change analyzerName to your analyzer's full computer name
Set app = CreateObject("AgilentPNA835x.Application", "analyzerName")

Set measData = app.ActiveMeasurement
Set chan = app.ActiveChannel

'To pick a location to get the data from remove the comment from one of these
dlocation = naRawData
'dlocation = naCorrectedData
'dlocation = naMeasResult
'dlocation = naRawMemory
'dlocation = naMemoryResult

'setup MSChart1 and MSChart2
'right click on the chart and select:
' - line chart
' - series in rows
End Sub
```

```
Sub GetComplex_Click()  
ReDim Data(numpts) As NAComplex  
Dim Real(201) AS Single  
Dim Imag(201) AS Single  
numpts = chan.NumberOfPoints
```

```
'You cannot change the format of Complex Data  
Call trigger  
'get data  
measData.GetNAComplex dlocation, numpts, Data(0)  
'plot data  
Dim i As Integer  
  
For i = 0 To numpts - 1  
    Real(i) = Data(i).Re  
    Imag(i) = Data(i).Im  
Next i  
MSChart1 = Real()  
MSChart2.Visible = True  
MSChart2 = Imag()  
Call Sweep  
End Sub
```

```
Sub GetPaired_Click()  
ReDim Real(numpts) As Single  
ReDim Imag(numpts) As Single  
numpts = chan.NumberOfPoints
```

```
' To pick a format, remove the comment from one of these  
fmt = naLogMagPhase  
'fmt = naLinMagPhase  
Call trigger  
'Get data  
measData.getPairedData dlocation, fmt, numpts, Real(0), Imag(0)  
'Plot Scalar  
MSChart1 = Real()  
MSChart2.Visible = True  
MSChart2 = Imag()  
Call Sweep  
End Sub
```

```
Sub GetScalar_Click()  
ReDim Data(numpts) As Single
```

```
numpts = chan.NumberOfPoints  
'To pick a format remove the comment from one of these  
fmt = naDataFormat_LogMag  
'fmt = naDataFormat_LinMag
```

```
'fmt = naDataFormat_Phase
'fmt = naDataFormat_Delay
'fmt = naDataFormat_Real
'fmt = naDataFormat_Imaginary
Call trigger
'Get data
measData.GetScalar dlocation, fmt, numpts, Data(0)
'Plot Data
MSChart1 = Data()
MSChart2.Visible = False
Call Sweep
End Sub
```

```
Sub trigger()
```

```
'The analyzer sends continuous trigger signals
app.TriggerSignal = naTriggerInternal
'The channel will only accept one, then go into hold
'Sync true will wait for the sweep to complete
```

```
sync=True
```

```
chan.Single sync
End Sub
```

```
Sub Sweep()
'The channel goes back to accepting all triggers
chan.Continuous
End Sub
```

Perform a Guided Calibration using COM

This example uses the [GuidedCalibration](#) interface to perform either a 2-port or 4-port calibration.

Learn more about [Reading and Writing Calibration data using COM](#).

See Other COM Example Programs

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save on the PNA hard drive as Calibrate.vbs. [Learn how to setup and run the macro](#).

```
Set pna = CreateObject("AgilentPNA835x.Application")
Set calMgr = pna.GetCalManager
Set guidedCal = calMgr.GuidedCalibration
Set chan = pna.ActiveChannel
chanNum = chan.ChannelNumber
' Initialize guided cal to be performed on the active channel.
' The boolean argument of True indicates to store the cal only
' in the channel's calibration register. If instead you wish
' to create a new calset that the new cal will get stored to,
' comment out this next line and uncomment the three lines below it.
guidedCal.Initialize chanNum, True
' Set calset = calMgr.CreateCalSet(chanNum)
' chan.SelectCalSet calset.GetGUID, True
' guidedCal.Initialize chanNum, False

' To perform 2-port cal, Uncomment the following
' Then comment the 4-port cal

' Do 2-port cal
' TwoPortGuidedCal

' Do 4-port cal
FourPortGuidedCal

Sub TwoPortGuidedCal()
' Select the connectors
guidedCal.ConnectorType(1) = "APC 3.5 female"
guidedCal.ConnectorType(2) = "APC 3.5 male"
For i = 3 To pna.NumberOfPorts
guidedCal.ConnectorType(i) = "Not used"
Next
value = MsgBox("Connectors defined for Ports 1 and 2")
' Select the Cal Kit for each port being calibrated.
guidedCal.CalKitType(1) = "85052D"
```

```

guidedCal.CalKitType(2) = "85052D"

' To use an ECal module instead, comment out the above two lines
' and uncomment the appropriate lines below:
' Your ECal module must already be connected
' via USB to the PNA.

'guidedCal.CalKitType(1) = "N4691-60004 ECal"
'guidedCal.CalKitType(2) = "N4691-60004 ECal"

' Non-factory characterizations are specified as follows:
'guidedCal.CalKitType(1) = "N4691-60004 User 1 ECal"

' When two or more ECal modules with the same model number are connected
' also specify the serial number as follows:
'guidedCal.CalKitType(1) = "N4691-60004 ECal 01234"

' When Disk Memory ECal user characterizations are used,
' specify both the User char and the serial number as follows:
'guidedCal.CalKitType(1) = "N4691-60004 MyDskChar ECal 01234"

MsgBox("Cal kits defined for Ports 1 and 2")

' Initiate the calibration and query the number of steps
numSteps = guidedCal.GenerateSteps
' Measure the standards, compute and apply the cal
MeasureAndComplete(numSteps)
End Sub

Sub FourPortGuidedCal()
' Select the connectors
guidedCal.ConnectorType(1) = "APC 3.5 female"
guidedCal.ConnectorType(2) = "APC 3.5 female"
guidedCal.ConnectorType(3) = "APC 3.5 female"
guidedCal.ConnectorType(4) = "APC 3.5 female"
' If a PNA which has more than 4 ports
For i = 5 To pna.NumberOfPorts
guidedCal.ConnectorType(i) = "Not used"
Next
value = MsgBox("Connectors defined for Ports 1 to 4")
' Select the Cal Kit for each port being calibrated.

```

```

guidedCal.CalKitType(1) = "85052D"
guidedCal.CalKitType(2) = "85052D"
guidedCal.CalKitType(3) = "85052D"
guidedCal.CalKitType(4) = "85052D"
' To use an ECal module instead, comment out the above four lines
' and uncomment the following four lines.
' Replace N4691-60003 with your own ECAL model followed by 'ECal'.
' Your ECal module must already be connected to a PNA USB port.
' See above for ECal options

'guidedCal.CalKitType(1) = "N4431-60003 ECal"
'guidedCal.CalKitType(2) = "N4431-60003 ECal"
'guidedCal.CalKitType(3) = "N4431-60003 ECal"
'guidedCal.CalKitType(4) = "N4431-60003 ECal"
value = MsgBox("Cal kits defined for Ports 1 to 4")
' Initiate the calibration
guidedCal.GenerateSteps
' If your selected cal kit is not a 4-port ECal module which can
' mate to all 4 ports at once, then you may want to choose which
' thru connections to measure for the cal. You must measure at
' least 3 different thru paths for a 4-port cal (for greatest
' accuracy you can choose to measure a thru connection for all 6
' pairings of the 4 ports). If you omit this command, the default
' is to measure from port 1 to port 2, port 1 to port 3, and
' port 1 to port 4. For this example we select to measure
' from port 1 to port 2, port 2 to port 3, and port 2 to port 4.
portList = Array(1,2,2,3,2,4)
guidedCal.ThruPortList = portList
' Re-generate the connection steps to account for the thru changes
numSteps = guidedCal.GenerateSteps
' Measure the standards, compute and apply the cal
MeasureAndComplete(numSteps)
End Sub

Sub MeasureAndComplete(ByVal numSteps)
value = MsgBox("Number of steps is " + CStr(numSteps))

' Measure the standards
'The following series of commands shows that standards
'can be measured in any order. These steps acquire
'measurement of standards in reverse order.
'It is easiest to iterate through standards using
'a For-Next Loop.
For i = NumSteps To 1
step = "Step " + CStr(i) + " of " + CStr(numSteps)
strPrompt = guidedCal.GetStepDescription(i)
value = MsgBox(strPrompt, vbOKOnly, step)
guidedCal.AcquireStep i
Next

' Conclude the calibration
guidedCal.GenerateErrorTerms

```

```
MsgBox ("Cal is done!")  
End Sub
```

Last Modified:

5-Apr-2011	Edited for ECal options
20-Jan-2007	Added any order to steps.

Perform a Source Power Cal using COM

This program can be run in either Visual Basic 6 or as a VBScript program. The PNA can run *.vbs programs as [macros](#).

This program demonstrates:

- Performing a source power calibration of Port 2 for Channel 1.
- Reading the calibration data.

Learn more about [Power Calibrations](#)

See an example that [Uploads a Source Power Cal](#)

See Other COM Example Programs

To run this program, you need:

- One of the following power meters connected to the PNA through GPIB: E4416A, E4417A, E4418A/B, E4419A/B, 437B, 438A, EPM-441A, EPM-442A

Note: If your power meter is other than these, you can [create your own Power Meter Driver](#) using our template.

- Your PC and PNA both connected to a LAN (for communicating with each other).

To make this program work in VBS, save the following code in a text editor file such as Notepad and save as *.vbs.

To make this program work in Visual Basic 6:

1. Create a new project
2. Click **Project, Add New Module**, click **Open**.
3. Paste the following code into the code window.
4. Delete the first two lines (comment and Main)
5. Click **Project, Properties**. Under **Startup Object**, select **Sub Main**
6. Click **Project, References**, and select the Agilent PNA Series Type Library.

```

' Run the Main subroutine
Main
Public Sub Main()
Dim PNA, chan, pwrCal
Const naPowerMeter = 0, naPowerMeterAndReceiver = 1
Const naPowerSensor_A = 0
Const naCorrectionValues = 0
Const port = 2
Const offset = 0
Const bDisplay = True
acquire
Dim stimulus, calvalues, strResult

' PNA COM objects
' enum NASourcePowerCalMethod
' enum NAPowerAcquisitionDevice
' enum NASourcePowerCalBuffer
' PNA port #2 as source port
' cal power offset value
' whether to display data during

' Instantiate our PNA COM objects
Set PNA = CreateObject("AgilentPNA835x.Application")
Set chan = PNA.Channels(1)
Set pwrCal = PNA.SourcePowerCalibrator

' Set the number of sweep points to 21 on Channel 1.
chan.NumberOfPoints = 21

' Specify the GPIB address of the power meter
' that will be used in performing the calibration.
pwrCal.PowerMeterGPIBAddress = 13

' Turn use of the loss table OFF (this assumes there is
' virtually no loss in the RF path to the power sensor
' due to a splitter, coupler or adapter).
pwrCal.UsePowerLossSegments = False

' Turn frequency checking OFF (so one power sensor is used for the entire cal
' acquisition sweep regardless of frequency span).
pwrCal.UsePowerSensorFrequencyLimits = False

' Specify a nominal power accuracy tolerance (IterationsTolerance) in dB for the
' calibration, and the maximum number of iterations to adjust power at each point,
' attempting to achieve within tolerance of the desired power. If at any stimulus
' point the power fails to reach within the set tolerance of the desired power
' after the maximum number of iterations, the power at that point will be set to the
' value determined by the last iteration (the Source Power Cal dialog box will
' indicate the FAIL, but we can still apply the cal if desired when it's complete).
' Each iteration is based upon a SETTLED power reading (see comments preceding the
' next two properties below).
pwrCal.IterationsTolerance = 0.1
pwrCal.MaximumIterationsPerPoint = 3

' The worst-case window of power uncertainty (for a calibration which meets
' tolerance) is the sum of the iteration tolerance and the power meter settling
' tolerance (which is described below).
' At each stimulus point, the PNA takes power meter readings and determines when
' they have settled by comparing the magnitude difference between consecutive
' readings versus a nominal dB tolerance limit (ReadingsTolerance) on that magnitude

```

```

' difference. When consecutive readings are within tolerance of each other, or
' if they are not within tolerance but we've taken a maximum number of readings
' (ReadingsPerPoint), the PNA does a weighted average of the readings taken at that
' stimulus point and that is considered our settled power reading.
pwrCal.ReadingsTolerance = 0.1
pwrCal.ReadingsPerPoint = 5

' Setup of information pertaining to this specific cal acquisition. Includes the
' method (type of devices) that will be used to perform the cal -- choose either
' naPowerMeter or naPowerMeterAndReceiver. naPowerMeterAndReceiver uses the power
' meter for the first iteration of each point and the PNA's reference receiver for
' subsequent iterations, so is much faster than using power meter only
naPowerMeter).
' But the power meter accounts for compression when calibrating at the output of an
' active device, whereas the reference receiver cannot unless it is coupled to the
' cal reference plane (on a PNA which allows direct access to the receivers).
' 'offset' specifies if the cal power level is offset (positive value for a gain,
' negative value for a loss) from the PNA port power setting on the channel when
' no source power cal is active. This is to account for components between the PNA
' test port and cal reference plane. In this example, we will calibrate at the PNA
' test port, so there is no offset (it is zero).
' 'bDisplay' indicates whether to display the source power cal dialog during the
' source power cal acquisition (the dialog will chart the corrected power readings).
pwrCal.SetCalInfo2 naPowerMeter, chan.ChannelNumber, port, offset, bDisplay

' Perform synchronous source power cal acquisition sweep using the sensor attached
' to Channel A of the power meter. This assumes that the power sensor is already
' connected to Port 2 of the PNA.
pwrCal.AcquirePowerReadings naPowerSensor_A, True

' Conclude the calibration. This applies the cal data to PNA channel memory,
' and turns the correction ON for Port 2 on Channel 1, but does NOT save the
' calibration.
pwrCal.ApplyPowerCorrectionValues

' At this point, if you choose to save the instrument state as a ".CST" file,
' the calibration will be saved with the instrument state in that file.
' Read the stimulus values from Channel 1.
stimulus = chan.GetXAxisValues

' Read the source power correction data.
calvalues = chan.getSourcePowerCalDataEx(naCorrectionValues, port)

' Print the data using a message box (here, Chr returns the ASCII characters
' for Tab (9) and Linefeed (10)).
strResult = "Stimulus" & Chr(9) & Chr(9) & "Cal Value" & Chr(10)
For i = 0 To UBound(stimulus)
strResult = strResult & stimulus(i) & Chr(9) & calvalues(i) & Chr(10)
Next
MsgBox strResult
End Sub

```

Perform an Unguided Cal using COM

This example uses the [ICalibrator](#) interface to do the following:

- perform a two port calibration
- retrieve the error term data
- retrieve the standard data (cal acquisition data)

Learn more about [Reading and Writing Calibration data using COM](#).

See Other COM Example Programs

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save on the PNA hard drive as Calibrate.vbs. [Learn how to setup and run the macro](#).

```
dim pna
' To run from an external PC, substitute your PNA Name and use the following
command.
' set pna = CreateObject("AgilentPNA835x.Application", "PNA Name")
set pna = CreateObject("AgilentPNA835x.Application")
dim calibrator
set calibrator=pna.activechannel.calibrator

wscript.echo "setcalinfo for two port cal"
calibrator.setcalinfo 5,1, 2

' only have one set of standards
calibrator.Simultaneous2PortAcquisition = false

'first acquire forward reflection standards, then reverse
dim p
for p = 1 to 2

if (p = 1) then
calibrator.AcquisitionDirection = 0
else
calibrator.AcquisitionDirection = 1
end if

wscript.echo "connect open to port ", p
calibrator.acquirecalstandard 1
```

```

wscript.echo "connect short to port ", p
calibrator.acquirecalstandard 2

wscript.echo "connect load to port ", p
calibrator.acquirecalstandard 3

next

wscript.echo "connect a thru1"
calibrator.acquirecalstandard 4

'Optional - perform isolation
wscript.echo "connect loads to both ports"
calibrator.acquirecalstandard 5

wscript.echo "calculating"
calibrator.CalculateErrorCoefficients

'Calibration complete
' Now read error terms and standard data

dim termName
termName= Array("Directivity","SourceMatch","ReflectionTracking")
dim vardata

' iterate over error terms
dim t
for t = 0 to 2 ' per error term
for p = 1 to 2 ' per port
wscript.echo "Requesting ",termName(t),p,p
vardata = calibrator.GetErrorTerm( t, p, p)
next
next

' now get the path terms: iterator each one request
termName = Array("Isolation", "LoadMatch", "TransmissionTracking")
for t = 0 to 2

wscript.echo "Requesting Forward term",termName(t),1,2
vardata = calibrator.GetErrorTerm( t, 1,2)
wscript.echo "Requesting Reverse Term",termName(t),2,1
vardata = calibrator.GetErrorTerm( t, 2,1)
next

dim stdname
stdname= Array("", "Open", "Short", "Load", "Thru", "Isolation")

' iterate over the port standards
for t = 1 to 3
for p = 1 to 2
' request the standard term for each port of interest

```

```
wscript.echo "Requesting",stdname(t),p,p  
  
vardata = calibrator.GetStandard( t, p, p)  
next  
next  
  
' now get the path standards: iterator each one request  
for t = 4 to 5  
  
wscript.echo "Requesting Forward",stdname(t),1,2  
vardata = calibrator.GetStandard( t, 1,2)  
  
wscript.echo "Requesting Reverse",stdname(t),2,1  
vardata = calibrator.GetStandard( t, 2,1)  
  
next
```

Perform an Unknown Thru or TRL Cal

The following program performs either a 2-port SOLT Unknown Thru Cal or a 2-port TRL Cal. The 85052C Cal Kit used in this program contains both types of standards. This program can be run on 2-port or 4-port PNAs. When run on [select PNA-L models](#), a Delta Match Cal is required. See [Delta Match Cal example program](#).

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Unknown.vbs. [Learn how to setup and run the macro](#).

```
PerformUnknownThruOrTRLCal
Sub PerformUnknownThruOrTRLCal()
' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
' Get the cal manager object
Set calMgr = app.GetCalManager
' Get the guided cal object
Set guidedCal = calMgr.GuidedCalibration
Set chan = app.ActiveChannel
chanNum = chan.ChannelNumber

' Initialize guided cal to be performed on the active channel.
' The boolean argument of True specifies the creation of a new calset
' for storing the new calibration.
guidedCal.Initialize chanNum, True

' Specify connectors for Ports 1 and 2
guidedCal.ConnectorType(1) = "APC 3.5 female"
guidedCal.ConnectorType(2) = "APC 3.5 male"

' If your PNA has more than 2 ports, uncomment one or both of
' these next two lines, to explicitly specify this is
' just a 2-port cal.
'guidedCal.ConnectorType(3) = "Not used"
'guidedCal.ConnectorType(4) = "Not used"

' Specify cal kit for Ports 1 and 2
guidedCal.CalKitType(1) = "85052C"
guidedCal.CalKitType(2) = "85052C"

' Since the 85052C cal kit contains SOLT standards and also TRL
' standards, these next lines determine whether the cal becomes
' unknown thru (SOLT), or TRL.
' Specify cal and Thru method

guidedCal.PathThruMethod (1,2) = "Undefined Thru"

' To set up the cal as TRL, comment the previous line and uncomment this next line.
```

The Thru method is set by default.

```
guidedCal.PathCalMethod (1,2) = "TRL"
```

```
'Always send Initialize after modifying the SmartCal logic
```

```
guidedCal.Initialize chanNum, True
```

```
numSteps = guidedCal.GenerateSteps
```

```
MsgBox "Number of steps is " + CStr(numSteps)
```

```
' Query the list of ports that need delta match
```

```
portList = guidedCal.PortsNeedingDeltaMatch
```

```
' If portList contains just one element and it's value is 0, then that indicates  
' none of the ports being calibrated require delta match data.
```

```
' If each testport on the PNA has it's own reference receiver (R channel),  
' then delta match is never needed, so portList will always be just 0.
```

```
lowerBound = LBound(portList)
```

```
If (UBound(portList) <> lowerBound) Or (portList(lowerBound) <> 0) Then
```

```
' Delta match data is required for at least one port.
```

```
' For this example, we assume a Global Delta Match Cal has previously been  
' performed so the Global Delta Match CalSet exists.
```

```
' Supplying an empty string to ApplyDeltaMatchFromCalSet indicates to use  
' the Global Delta Match CalSet.
```

```
guidedCal.ApplyDeltaMatchFromCalSet ""
```

```
End If
```

```
' Measure the standards
```

```
For i = 1 To numSteps
```

```
step = "Step " + CStr(i) + " of " + CStr(numSteps)
```

```
strPrompt = guidedCal.GetStepDescription(i)
```

```
retVal = MsgBox(strPrompt, vbOKCancel, step)
```

```
If retVal = vbCancel Then Exit Sub
```

```
guidedCal.AcquireStep i
```

```
Next
```

```
' Conclude the calibration
```

```
guidedCal.GenerateErrorTerms
```

```
MsgBox "Cal is done!"
```

```
End Sub
```

Perform Global Delta Match Cal

The following program performs a [Global Delta Match Calibration](#). This is required when performing an Unknown Thru cal or TRL cal on PNAs without a reference receiver for each test port. [See example](#).

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Delta.vbs. [Learn how to setup and run the macro](#).

```
Sub PerformGlobalDeltaMatchCal()  
  ' Create / Get the PNA application.  
  Set app = CreateObject("AgilentPNA835x.Application")  
  ' Get cal manager object  
  Set calMgr = app.GetCalManager  
  ' Get guided cal object  
  Set guidedCal = calMgr.GuidedCalibration  
  
  ' Initiate a Global Delta Match calibration, choosing connector and cal kit  
  numSteps = guidedCal.GenerateGlobalDeltaMatchSequence("APC 3.5 female", "85033D/E")  
  MsgBox "Number of steps is " + CStr(numSteps)  
  
  ' Measure the standards  
  For i = 1 To numSteps  
    step = "Step " + CStr(i) + " of " + CStr(numSteps)  
    strPrompt = guidedCal.GetStepDescription(i)  
    retVal = MsgBox(strPrompt, vbOKCancel, step)  
    If retVal = vbCancel Then Exit Sub  
    guidedCal.AcquireStep i  
  Next  
  
  ' Conclude the calibration  
  guidedCal.GenerateErrorTerms  
  MsgBox "Cal is done!"  
  
End Sub
```

Perform a Guided Cal with C#

The following example performs a 2-port or 4-port Guided Cal using C#.

Note: Replace **<remote host name>** with the full computer name of your PNA.

[Learn more about using .NET with the PNA](#)

```
AgilentPNA835x.IApplication pna;

    AgilentPNA835x.ICalManager3 calMgr;

    AgilentPNA835x.IGuidedCalibration guidedCal;

    AgilentPNA835x.IChannel chan;

    int chanNum;

    AgilentPNA835x.ICalSet calset;

    void PerformGuidedCal()
    {
        Type pnaType = Type.GetTypeFromProgID("AgilentPNA835x.Application",
"<remote host name>");

        pna = (AgilentPNA835x.IApplication)
            Activator.CreateInstance(pnaType);

        calMgr = (AgilentPNA835x.ICalManager3)pna.GetCalManager();

        guidedCal =
(AgilentPNA835x.IGuidedCalibration)calMgr.GuidedCalibration;

        chan = pna.ActiveChannel;

        chanNum = chan.channelNumber;

        // Initialize guided cal to be performed on the active channel.
        // The boolean argument of True indicates to store the cal only
        // in the channel's calibration register. If instead you wish
        // to create a new calset that the new cal will get stored to,
        // comment out this next line and uncomment the three lines below it.
```

```
guidedCal.Initialize(chanNum, true);

calset = calMgr.CreateCalSet(chanNum);

chan.SelectCalSet(calset.GetGUID(), true);

guidedCal.Initialize(chanNum, false);

// To perform 2-port cal, Uncomment the following
// Then comment the 4-port cal

// Do 2-port cal
// TwoPortGuidedCal();

// Do 4-port cal
FourPortGuidedCal();
}

void TwoPortGuidedCal()
{
// Select the connectors

    guidedCal.set_ConnectorType(1, "APC 3.5 female");
guidedCal.set_ConnectorType(2, "APC 3.5 male");
for (int i = 3; i <= pna.NumberOfPorts; i++)
    guidedCal.set_ConnectorType(i, "Not used");
MessageBox.Show("Connectors defined for Ports 1 and 2");

// Select the Cal Kit for each port being calibrated.
guidedCal.set_CalKitType(1, "85052D");
guidedCal.set_CalKitType(2, "85052D");

// To use an ECal module instead, comment out the above two lines
```

```
// and uncomment the following two lines.

// Replace N4691-60004 with your own ECAL model followed by 'ECal'.
// Your ECal module must already be connected to a PNA USB port.
// guidedCal.CalKitType(1) = "N4691-60004 ECal"
// guidedCal.CalKitType(2) = "N4691-60004 ECal"
MessageBox.Show("Cal kits defined for Ports 1 and 2");

// Initiate the calibration and query the number of steps
int numSteps = guidedCal.GenerateSteps();

// Measure the standards, compute and apply the cal
MeasureAndComplete(numSteps);
}

void FourPortGuidedCal()
{
    //Select the connectors
    guidedCal.set_ConnectorType(1,"APC 3.5 female");
    guidedCal.set_ConnectorType(2,"APC 3.5 female");
    guidedCal.set_ConnectorType(3,"APC 3.5 female");
    guidedCal.set_ConnectorType(4,"APC 3.5 female");

    // If a PNA which has more than 4 ports
    for (int i = 5;i<=pna.NumberOfPorts;++i)
        guidedCal.set_ConnectorType(i,"Not used");

    MessageBox.Show("Connectors defined for Ports 1 to 4");

    // Select the Cal Kit for each port being calibrated.
    guidedCal.set_CalKitType(1,"85052D");
    guidedCal.set_CalKitType(2,"85052D");
    guidedCal.set_CalKitType(3,"85052D");
```

```
guidedCal.set_CalKitType(4, "85052D");

// To use an ECal module instead, comment out the above four lines
// and uncomment the following four lines.
// Replace N4691-60003 with your own ECAL model followed by 'ECal'.
// Your ECal module must already be connected to a PNA USB port.
//guidedCal.CalKitType(1) = "N4431-60003 ECal";
//guidedCal.CalKitType(2) = "N4431-60003 ECal";
//guidedCal.CalKitType(3) = "N4431-60003 ECal";
//guidedCal.CalKitType(4) = "N4431-60003 ECal";

MessageBox.Show("Cal kits defined for Ports 1 to 4");

// Initiate the calibration
guidedCal.GenerateSteps();

// If your selected cal kit is not a 4-port ECal module which can
// mate to all 4 ports at once, then you may want to choose which
// thru connections to measure for the cal. You must measure at
// least 3 different thru paths for a 4-port cal (for greatest
// accuracy you can choose to measure a thru connection for all 6
// pairings of the 4 ports). If you omit this command, the default
// is to measure from port 1 to port 2, port 1 to port 3, and
// port 1 to port 4. For this example we select to measure
// from port 1 to port 2, port 2 to port 3, and port 2 to port 4.

long[] portList = new long[6]{1,2,2,3,2,4};

guidedCal.ThruPortList = portList;

// Re-generate the connection steps to account for the thru changes
int numSteps = guidedCal.GenerateSteps();

// Measure the standards, compute and apply the cal
MeasureAndComplete(numSteps);

}
```

```
void MeasureAndComplete(int numSteps)
{
    MessageBox.Show("Number of steps is " + numSteps.ToString());

    // Measure the standards
    // The following series of commands shows that standards
    // can be measured in any order. These steps acquire
    // measurement of standards in reverse order.
    // It is easiest to iterate through standards using
    // a For-Next Loop.
    for (int i = numSteps; i >= 1; --i)
    {
        string strPrompt = guidedCal.GetStepDescription(i);
        MessageBox.Show(strPrompt);
        guidedCal.AcquireStep(i);
    }

    // Conclude the calibration
    guidedCal.GenerateErrorTerms();
    MessageBox.Show("Cal is done!");
}
```

Last Modified:

27-Jan-2009 New topic.

Perform an ECal using COM

This example uses the [GuidedCalibration](#) interface to perform a 2-port ECal calibration.

Learn more about [Reading and Writing Calibration data using COM](#).

See Other COM Example Programs

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save on the PNA hard drive as ECal.vbs. [Learn how to setup and run the macro](#).

```
Set pna = CreateObject("AgilentPNA835x.Application")
Set calMgr = pna.GetCalManager
Set guidedCal = calMgr.GuidedCalibration
Set chan = pna.ActiveChannel
chanNum = chan.ChannelNumber
' Initialize guided cal to be performed on the active channel.
' The boolean argument of True indicates to create a new calset
guidedCal.Initialize chanNum, True
' To perform 3-port cal, Uncomment the following
' Then comment the 2-port cal

' Do 2-port cal
TwoPortGuidedCal

' Do 3-port cal
' ThreePortGuidedCal

Sub TwoPortGuidedCal()
'Change the following to match the connectors on your ECal module
guidedCal.ConnectorType(1) = "APC 3.5 female"
guidedCal.ConnectorType(2) = "APC 3.5 female"
For i = 3 To pna.NumberOfPorts
guidedCal.ConnectorType(i) = "Not used"
Next
value = MsgBox("Connectors defined for Ports 1 and 2")

' Select the ECal module for each port being calibrated.
' Replace N4691-60004 with your own ECAL model followed by 'ECAL'.
' Your ECal module must already be connected
' via USB to the PNA.
```

```

guidedCal.CalKitType(1) = "N4691-60004 ECal"
guidedCal.CalKitType(2) = "N4691-60004 ECal"
' Non-factory characterizations are specified as follows:
'guidedCal.CalKitType(1) = "N4691-60004 User 1 ECal"
' When two or more ECal modules with the same model number are connected
' also specify the serial number as follows:
'guidedCal.CalKitType(1) = "N4691-60004 ECal 01234"
' When Disk Memory ECal user characterizations are used,
' specify both the User char and the serial number as follows:
'guidedCal.CalKitType(1) = "N4691-60004 MyDskChar ECal 01234"
' Turn on auto orientation for the ECal (default behavior).
'guidedCal.AutoOrient = 1'

MsgBox("Cal kits defined for Ports 1 and 2")

' Initiate the calibration and query the number of steps
numSteps = guidedCal.GenerateSteps
' Measure the standards, compute and apply the cal
MeasureAndComplete(numSteps)
End Sub

Sub ThreePortGuidedCal()
'Change the following to match the connectors on your ECal module
guidedCal.ConnectorType(1) = "APC 3.5 female"
guidedCal.ConnectorType(2) = "APC 3.5 female"
guidedCal.ConnectorType(3) = "APC 3.5 female"

' Select the ECal module for each port being calibrated.
' Replace N4691-60003 with your own ECAL model followed by 'ECal'.
' Your ECal module must already be connected to a PNA USB port.
guidedCal.CalKitType(1) = "N4431-60003 ECal"
guidedCal.CalKitType(2) = "N4431-60003 ECal"
guidedCal.CalKitType(3) = "N4431-60003 ECal"

value = MsgBox("Cal kits defined for Ports 1 to 3")
' Initiate the calibration
numSteps = guidedCal.GenerateSteps
' Measure the standards, compute and apply the cal
MeasureAndComplete(numSteps)
End Sub

```

```
Sub MeasureAndComplete(ByVal numSteps)
value = MsgBox("Number of steps is " + CStr(numSteps))
' Measure the standards
For i = 1 To numSteps
step = "Step " + CStr(i) + " of " + CStr(numSteps)
strPrompt = guidedCal.GetStepDescription(i)
value = MsgBox(strPrompt, vbOKOnly, step)
guidedCal.AcquireStep i
Next

' Conclude the calibration
guidedCal.GenerateErrorTerms
MsgBox ("Cal is done!")
End Sub
```

Last Modified:

5-Apr-2011 Edited for ECal options

Perform an ECal User Characterization

This example performs a user-characterization and stores it to both the ECal module memory and PNA disk memory.

It then performs two 2-port cals: the first using the characterization from module memory, then using the characterization from disk memory.

Note: This example requires that channel 1 be already calibrated.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the PNA hard drive as ECal.vbs.

See Also

[How to setup and run the macro.](#)

[ECalUserCharacterizer Object](#)

[About User Characterization](#)

See Other COM Example Programs

```
Option Explicit

Dim pna

Set pna = CreateObject("AgilentPNA835x.Application")

Dim calMgr

Set calMgr = pna.GetCalManager

' Get ECal User Characterizer COM object

Dim ecalCharacterizer

Set ecalCharacterizer = calMgr.GetECalUserCharacterizer

' Substitute here the model number and serial number of your own ECal.

' Note that this example corresponds to a 4-port ECal module with

' serial number 00001. If instead you have a 2-port ECal module,

' their model numbers are '5x5' numbers -- for example, 'N4691-60001'.

Dim ecalModelNum
```

```
ecalModelNum = "N4433A"

Dim ecalSerialNum

ecalSerialNum = "00001"

ecalCharacterizer.ECalID = ecalModelNum & "," & ecalSerialNum

MsgBox "ECal module to be characterized is: " & ecalCharacterizer.ECalID

' Set which user characterization number (1-12) the new characterization
' will be stored to in the ECal module when it is done. If you intend to
' store your user characterization just to PNA Disk Memory and NOT the
' ECal module's memory, then omit the setting of this property.

ecalCharacterizer.CharacterizationNumber = 1

' The following commented-out lines of code show how you can access
' the list of connector type names you can set for the ports of an
' ECal when you user-characterize it. However, please note that if
' you are writing the user characterization to the ECal module's memory,
' as of yet only the Factory Defined set of connector choices will work
' properly (see the ValidConnectorType property).

' If you will be saving your characterization to just
' PNA Disk Memory only, then all connector names returned by this call
' will work, user-defined connector names as well as factory-defined.

'Dim connTypeArray

'connTypeArray = ecalCharacterizer.ValidConnectorTypes

'MsgBox connTypeArray(1)

' Access element 1 in the string array

' For each port of the ECal module, specify which connector type
' is at the end of the adapter (or cable or fixture) that is
' connected to that port of the ECal for the characterization
' (must be one of the connector types that is included in the
' list that the ValidConnectorTypes method returns). The default
```

```
' is "No adapter", which assumes you are characterizing that port
' of the ECal "as is" (nothing attached to it). So in this example,
' Ports C and D of the ECal are being characterized to just the
' ECal's connectors.

ecalCharacterizer.ConnectorType(1) = "APC 3.5 male" ' ECal Port A
ecalCharacterizer.ConnectorType(2) = "APC 3.5 male" ' ECal Port B

' As with the connector types, the information set in these next
' few properties also gets stored within the characterization.
' Set the name of the person and/or company that is producing
' this characterization.

ecalCharacterizer.UserName = "John Doe, Acme Inc."

' Set user-specified description of the PNA being used.

ecalCharacterizer.UserDescriptionOfPNA = "SN US12345678"

' Set descriptions of what you have connected to the ECal module's
' ports for the characterization.

ecalCharacterizer.PortDescription(1) = "3.5 mm adapter, SN 00001" ' Port A of the
Ecal
ecalCharacterizer.PortDescription(2) = "3.5 mm adapter, SN 00002" ' Port B of the
Ecal

' Begin a user characterization on Channel 1.

' If you will be storing this characterization to the ECal module's memory, then
' the boolean argument to this command should be set to True. If you will be
storing
' this characterization to PNA disk memory ONLY, then you should specify False
for
' that argument. In this example we will be storing the characterization to both
' module memory and PNA disk memory, so we use True.

ecalCharacterizer.InitializeEx 1, True

' Generate the measurement steps for the user characterization.
```

```
Dim numSteps

numSteps = ecalCharacterizer.GenerateSteps

' Measure the steps.

' You must ensure you have already applied the appropriate calibration to the
channel

' already, or else an error will be thrown indicating that.

Dim i

For i = 1 To numSteps

    MsgBox ecalCharacterizer.GetStepDescription(i)

    ecalCharacterizer.AcquireStep(i)

    MsgBox "Acquire is complete"

Next

MsgBox "Now the user characterization will be saved to the ECal module and to PNA
disk memory"

' Save the user characterization to the ECal module's memory.

' Note that this can take multiple minutes, depending on how

' many sweep points the channel has.

ecalCharacterizer.SaveToECal

' Save the user characterization to PNA Disk Memory.

Dim characterizationName

characterizationName = "test"

ecalCharacterizer.SaveToDiskMemory(characterizationName)

MsgBox "User characterization is complete. Now we will calibrate using it.
First we will use it from ECal module memory."

Dim moduleMemCalKitName

moduleMemCalKitName = GetCalKitName("User " &
CStr(ecalCharacterizer.CharacterizationNumber))

DoTwoPortCal moduleMemCalKitName

MsgBox "Now we will calibrate using the characterization from PNA Disk Memory."

Dim pnaDiskMemCalKitName
```

```

pnaDiskMemCalKitName = GetCalKitName(characterizationName)

DoTwoPortCal pnaDiskMemCalKitName

MsgBox "Example has completed"

Function GetCalKitName(characterizationName)

Dim calKitName

calKitName = ecalModelNum

If Len(characterizationName) > 0 Then calKitName = calKitName & " " &
characterizationName

calKitName = calKitName & " ECal " & ecalSerialNum

GetCalKitName = calKitName

End Function

Sub DoTwoPortCal(calKitName)

' Initialize guided cal to be performed on Channel 1.

Dim guidedCal

Set guidedCal = calMgr.GuidedCalibration

guidedCal.Initialize 1, True

' Specify the DUT connector for each PNA port to be calibrated (DUT connector =
ECal characterization's connector)

guidedCal.ConnectorType(1) = "APC 3.5 male"

guidedCal.ConnectorType(2) = "APC 3.5 male"

' Specify the "cal kit" for each of those ports

guidedCal.CalKitType(1) = calKitName

guidedCal.CalKitType(2) = calKitName

' We know this example will result in a calibration sequence of a single
"connection step"

Dim numSteps

numSteps = guidedCal.GenerateSteps

```

```
' Acquire the cal connection step
```

```
guidedCal.AcquireStep 1
```

```
' Conclude the cal and turn it on
```

```
guidedCal.GenerateErrorTerms
```

```
End Sub
```

Last Modified:

18-Sep-2009 Updated with new User Char commands (9.0)

10-Nov-2008 New topic (8.33)

Perform a Comprehensive 2-Port Guided Cal

This example program performs a Guided Calibration on the active channel between ports 1 and 2. The following calibration features are demonstrated:

- Guided Power Cal
- Optional functions when using ECal
- Select the Thru method
- Save to a new CalSet

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the PNA hard drive as CompGuided.vbs. Learn [How to setup and run the macro.](#)

See Also

[PNA Object Model](#)

[CalManager Object](#)

[GuidedCalibration Object](#)

[Calibrator Object](#)

See Other COM Example Programs

```
' Performing a Guided 2-port cal (Ports 1 and 2)

TwoPortGuidedCal

Sub TwoPortGuidedCal

' Create / Get the PNA application.

Set app = CreateObject("AgilentPNA835x.Application")

Set calMgr = app.GetCalManager

Set guidedCal = calMgr.GuidedCalibration

Set chan = app.ActiveChannel

chanNum = chan.ChannelNumber
```

```

' Initialize guided cal to be performed on the active channel.
' The boolean argument of True indicates to create a new calset
' for storing the new calibration to.
guidedCal.Initialize chanNum, True

' Query the connectors that the PNA system recognizes
conns = guidedCal.ValidConnectorTypes

' Format the list string with linefeed characters between each substring
connList = FormatList(conns)

' Select the connector for Port 1
selectedConn1 = InputBox("Enter your DUT connector for Port 1. Choose from this
list:" & _
                        Chr(10) & Chr(10) & connList)

If selectedConn1 = "" Then Exit Sub
guidedCal.ConnectorType(1) = selectedConn1

' Select the connector for Port 2
selectedConn2 = InputBox("Enter your DUT connector for Port 2. Again, choose
from this list:" & _
                        Chr(10) & Chr(10) & connList)

If selectedConn2 = "" Then Exit Sub
guidedCal.ConnectorType(2) = selectedConn2

' Note: If your PNA has more than 2 ports, you would need to uncomment
' one or both of these next two lines, to explicitly specify this is
' just a 2-port cal.
'guidedCal.ConnectorType(3) = "Not used"
'guidedCal.ConnectorType(4) = "Not used"

' Query the list of acceptable cal kits and ECal module characterizations for
Port 1.
kits = guidedCal.GetCompatibleCalKits(selectedConn1)

' Format the list string with linefeed characters between each substring

```

```

kitList = FormatList(kits)

' Select the Cal Kit or ECal module characterization to use for Port 1.

selectedKit = InputBox("Enter your cal kit or ECal module characterization for
Port 1. " & _

                        "Choose from this list:" & Chr(10) & Chr(10) & kitList)

If selectedKit = "" Then Exit Sub

guidedCal.CalKitType(1) = selectedKit

' Query the list of acceptable cal kits and ECal module characterizations for
Port 2.

kits = guidedCal.GetCompatibleCalKits(selectedConn2)

' Format the list string with linefeed characters between each substring

kitList = FormatList(kits)

' Select the Cal Kit or ECal module characterization to use for Port 2.

selectedKit = InputBox("Enter your cal kit or ECal module characterization for
Port 2. " & _

                        "Choose from this list:" & Chr(10) & Chr(10) & kitList)

If selectedKit = "" Then Exit Sub

guidedCal.CalKitType(2) = selectedKit

' This determines whether the cal will be a "Guided Power Cal"

' or just a traditional S-parameter cal.

message = "On which port number shall power be measured? "

message = message & "For a traditional guided cal without power cal, enter 0"

Dim powerPort

powerPort = CInt( InputBox(message) )

If powerPort > 0 Then

    guidedCal.PerformPowerCalibration(powerPort) = True

    Dim retVal

    retVal = MsgBox("Is the power sensor's connector type or gender different from
the DUT connector for that port?", vbYesNo)

    If retVal = vbYes Then

```

```

message = "Enter your power sensor's connector. Choose from this list:"
message = message & Chr(10) & Chr(10) & connList
' Select the sensor's connector.
selectedConn1 = InputBox(message)
If selectedConn1 = "" Then Exit Sub
guidedCal.PowerSensorConnectorType(powerPort) = selectedConn1
' Query the list of acceptable cal kits and ECal module characterizations
' that are applicable for the sensor's connector.
kits = guidedCal.GetCompatibleCalKits(selectedConn1)
' Format the list string with linefeed characters between each substring
kitList = FormatList(kits)
message = "Enter your cal kit or ECal module characterization to use for
de-embed of the sensor's connector. "
message = message & "Choose from this list:"
message = message & Chr(10) & Chr(10) & kitList
' Select the Cal Kit or ECal module characterization to use for de-embed of
the sensor's connector.
selectedKit = InputBox(message)
If selectedKit = "" Then Exit Sub
guidedCal.PowerSensorCalkitType(powerPort) = selectedKit
Else
guidedCal.PowerSensorConnectorType(powerPort) = "Ignored"
End If ' End of block that considers the sensor's connector
' Ask for the power level to perform the power cal at
' (if this command is omitted, the default is 0 dBm).
Dim powerLevel
powerLevel = InputBox("Enter the power level for the power cal to be performed
at")
If powerLevel = "" Then Exit Sub

```

```

    guidedCal.PowerCalibrationPowerLevel(powerPort) = CDb1(powerLevel)

Else

    guidedCal.PerformPowerCalibration(1) = False

End If ' End of block that considers if the cal will include power calibration

'-----
'---

' This next block of commented-out code shows optional functions when using ECal.
' These OrientECALModule and ECALPortMapEx properties would need to be set prior
to
' calling GenerateSteps on the guidedCal object.
' Read the information about the Agilent factory characterization data
' of ECal module #1 on the USB bus
'Set calibrator = chan.Calibrator
'Const ECalModule1 = 1
'module1Info = calibrator.GetECALModuleInfoEx(ECalModule1)
'MsgBox "Description of ECal module #1:" & Chr(10) & Chr(10) & module1Info
' By default, during calibration the PNA automatically determines the orientation
of
' the ECal module (senses which port of the module is connected to which port of
the PNA).
' However, since this setting could have recently been overridden by another user
of
' the instrument, use this next line to ensure the auto orientation setting is
enabled.

'calibrator.OrientECALModule = True

' Alternatively, if you are measuring at very low power levels where
' the PNA fails to sense the module's orientation, you may need to turn off the
auto
' orientation and specify how the module is connected (as in these next two lines
of code,
' "A1,B2" would indicate Port A of the module is connected to Port 1 and
' Port B is connected to Port 2).

```

```

'calibrator.OrientECALModule = False

'calibrator.ECALPortMapEx(ECalModule1) = "A1,B2"

' End of optional ECal setup

'-----
---

' Select the thru method of Default. This instructs the PNA to determine which
thru

' standard measurement technique to use, based upon the selected connectors and
' calibration kit(s) and what model of PNA this is.

guidedCal.ThruCalMethod = 0 ' 0 = naDefaultCalMethod

' Initiate the calibration and query the number of steps

numSteps = guidedCal.GenerateSteps

MsgBox "Number of steps is " + CStr(numSteps)

' Measure the standards

For i = 1 To numSteps

    step = "Step " + CStr(i) + " of " + CStr(numSteps)

    strPrompt = guidedCal.GetStepDescription(i)

    MsgBox strPrompt, vbOKOnly, step

    guidedCal.AcquireStep i

Next

' Conclude the calibration

guidedCal.GenerateErrorTerms

MsgBox "Cal is done!"

End Sub

Function FormatList(tokens)

    For i = 0 To UBound(tokens)

        list = list & tokens(i) & Chr(10)

```

Next

FormatList = list

End Function

Last Modified:

8-Oct-2010 MX New topic

ECAL Confidence Check

This Visual Basic program:

- Initializes the PNA objects.
- Performs a complete ECAL confidence check

Before using this code:

- The active channel must contain an S11 measurement with a 1-port or N-port calibration
- Prepare a form with two buttons named **cmdRun** and **cmdQuit**

Note: A confidence check can NOT be performed remotely from User Characterizations that are stored on the PNA disk.

```
Private oPNA As AgilentPNA835x.Application
Private oChan As Channel
Private oCal As Calibrator
Private oMeas As Measurement

Private Sub cmdRun_Click()
Dim iMeasIndex As Integer

Set oPNA = CreateObject("AgilentPNA835x.Application", "MachineName")
Set oChan = oPNA.ActiveChannel
Set oCal = oChan.Calibrator

iMeasIndex = 1

' Loop through measurements until an S11 on the active channel
' is found, or the end of the measurement collection is reached.
Do
    Set oMeas = oPNA.Measurements(iMeasIndex)
```

```

If oMeas.Parameter = "S11" And _
    oMeas.channelNumber = oChan.channelNumber Then Exit Do
iMeasIndex = iMeasIndex + 1
If iMeasIndex > oPNA.Measurements.Count Then
    MsgBox "No S11 measurement found on the active channel." _
        & " Create an S11 measurement, then try again."
    Exit Sub
End If
Loop

' Set up trace view so we are viewing only the data trace.
oMeas.View = naData
' Acquire the S11 confidence check data from ECal Module A
' into the memory buffer.
oCal.AcquireCalConfidenceCheckECALEx "S11", 1
' Turn on trace math so the trace shows data divided by memory.
' You can be confident the S11 calibration is reasonably good if
' the displayed trace varies no more than a few tenths of a dB
' from 0 dB across the entire span.
oMeas.TraceMath = naDataDivMemory
End Sub

Sub cmdQuit_Click()
' Turn off trace math
' in case someone clicks Quit without having clicked Run
If oMeas <> Nothing Then oMeas.TraceMath = naDataNormal
' Conclude the confidence check to set the ECal module
' back to it's idle state.

```

```
If oCal <> Nothing Then oCal.DoneCalConfidenceCheckECAL
```

```
' End the program
```

```
End
```

```
End Sub
```

Writing Cal Set Data using COM

This example creates a Cal Set and then writes data to the Cal Set.

Learn more about [Reading and Writing Calibration data using COM](#).

Learn more about [Cal Sets](#).

See example: [Reading Calset Data](#)

See Other COM Example Programs

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save on the PNA hard drive as CalSetsWrite.vbs. [Learn how to setup and run the macro](#).

```
dim pna
dim v
Set pna=CreateObject("AgilentPNA835x.Application")
InitPhonyData
PutPhonyData
' This sub creates phony data
Sub InitPhonyData()
Dim i
Dim numpts
wscript.echo "init phony"
numpts = pna.ActiveChannel.NumberOfPoints
ReDim v(numpts - 1, 1)
For i = 0 To numpts - 1
v(i, 0) = i
v(i, 1) = 0
Next
End Sub

' This sub creates a Cal Set, then writes the phony data to it
Sub PutPhonyData()
Dim cmgr
Dim cset
wscript.echo "putphony"
Set cmgr = pna.GetCalManager
Set cset = cmgr.CreateCalSet(1)
cset.OpenCalSet naCalType_OnePort, 1
const directivity = 0
const sourcematch=1
const reflectiontracking =2
```

```
cset.putErrorTerm directivity,1, 1, v
cset.putErrorTerm sourcematch,1, 1, v
cset.putErrorTerm reflectiontracking ,1, 1, v
cset.CloseCalSet
cset.Description = "Phony One Port"
cset.save
End Sub
```

Upload a Source Power Cal using COM

This program can be run in either Visual Basic 6 or as a VBScript program. The PNA can run *.vbs programs as [macros](#).

This program demonstrates:

- Uploading a source power calibration of Port 2 for Channel 1.
- Reading the calibration data.

Learn more about [Power Calibrations](#)

See Other COM Example Programs

To run this program you need:

- Your PC and PNA both connected to a LAN (for communicating with each other).

To make this program work in VBS, save the following code in a text editor file such as Notepad and save as *.vbs.

To make this program work in Visual Basic 6:

1. Create a new project
2. Click **Project, Add New Module**, click **Open**.
3. Paste the following code into the code window.
4. Delete the first two lines (comment and Main)
5. Click **Project, Properties**. Under **Startup Object**, select **Sub Main**
6. Click **Project, References**, and select the Agilent PNA Series Type Library.

```
' Run the Main subroutine
Main
Public Sub Main()
Dim PNA, chan           ' PNA COM objects
Const naCorrectionValues = 0 ' enum NASourcePowerCalBuffer
Const port = 2         ' PNA port #2 as source port
Dim stimulus, calvalues
```

```

Dim power, calpower, strResult
' Instantiate our PNA COM objects
Set PNA = CreateObject("AgilentPNA835x.Application")
Set chan = PNA.Channels(1)

' Set the number of sweep points to 2 on Channel 1.
chan.NumberOfPoints = 2

' Ensure there's currently no source power cal on for this channel and port.
chan.SourcePowerCorrection(port) = False

' Specify if the cal power level is offset (positive value for a gain, negative
' value for a loss) from the PNA port power setting on the channel when
' no source power cal is active. This is to account for components
' between the PNA test port and cal reference plane.
' In this example, let's set up our calibration
' at the output of an amplifier with 15 dB gain.
chan.SourcePowerCalPowerOffset(port) = 15

' Send our source power correction data to the PNA. For purpose of simplicity
' in this example, we'll set up for no correction (0) at our start stimulus and
' 0.5 dB at our stop stimulus (recall that our sweep currently has just 2 points).
calvalues = Array(0, 0.5)
chan.putSourcePowerCalDataEx naCorrectionValues, port, calvalues

' Set the number of sweep points to 21 on Channel 1.
chan.NumberOfPoints = 21

' Read the fixed power level for this port on Channel 1.
power = chan.TestPortPower(port)

' Turn the source power cal on.
chan.SourcePowerCorrection(port) = True

' Again read the fixed power level for this port on Channel 1
' (with our calibration turned on, this should now include the 15 dB offset
' we indicated our power amplifier provides).
calpower = chan.TestPortPower(port)

' Read the stimulus values from Channel 1.
stimulus = chan.GetXAxisValues

' Read back the source power correction data, now interpolated for 21 points
calvalues = chan.getSourcePowerCalDataEx(naCorrectionValues, port)

' Print the data using a message box (here, Chr returns the ASCII characters
' for Tab (9) and Linefeed (10)).
strResult = "PNA port power = " & power & Chr(10)
strResult = strResult & "Power at reference plane = " & calpower & Chr(10) & Chr(10)
strResult = strResult & "Stimulus" & Chr(9) & Chr(9) & "Cal Value" & Chr(10)
For i = 0 To UBound(stimulus)
strResult = strResult & stimulus(i) & Chr(9) & calvalues(i) & Chr(10)

```

```
Next  
MsgBox strResult  
End Sub
```

Upload and Download Segment Table

These example programs use the [SetAllSegments_Method](#) and [GetAllSegments_Method](#) to do the following:

- Creates a 2-dimensional array (7 x 10) 7 data elements that define each segment x 10 segments
- Uploads the data to the PNA
- [Downloads a segment table](#) from the PNA

This program does not make sweep type = segment or show the segment table.

The comments indicate the order in which the segment elements are specified: Index 0 - segment state, Index 4 is IFBW, and so forth.

See Other COM Example Programs

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save on the PNA hard drive as *.vbs. [Learn how to setup and run the macro.](#)

```
' Create the application instance, and preset the application
Set app = CreateObject("AgilentPNA835x.Application")
app.Preset

Dim chan
Set chan = app.ActiveChannel
chan.sweepType = 4

Dim segs
Set segs = chan.Segments

Dim win
Set win = app.NAWindows(1)
win.ShowTable 2

' Multipliers
kHz = 1000
MHz = kHz*1000
GHz = MHz*1000
' Create segments from 10MHz to 3GHz
StartFreq = 10 * MHz
```

```

StopFreq = 3 * GHz
'*
'* Create 10 segments between StartFreq and StopFreq
'*
' Create a 2-D array of segments.
' 1st dimension is size 7 (6 is max index)
' to hold all the data per segment.
' 2nd dimension is size 10 (9 is max index)
' to hold 10 total segments.
Dim segdata(6, 9)
' Width of frequency segment, used below
SegmentWidth = (StopFreq-StartFreq)/10
' Fill up all 10 segments (indicies 0 to 9) with data
For i = 0 To 9
' element 0=segment state (on or off)
segdata(0, i) = True

' element 1=Num Points in this segment
segdata(1, i) = 500

' element 2=Start Freq
segdata(2, i) = StartFreq + i * SegmentWidth

' element 3=Stop Freq
segdata(3, i) = segdata(2, i) + SegmentWidth

' element 4=IFBW
segdata(4, i) = 35000

' element 5=Dwell Time
segdata(5, i) = 0

' element 6=Power
segdata(6, i) = 0

Next

' Configure Independent segment settings
segs.IFBandwidthOption = 1
segs.SourcePowerOption = 1

' Push the segment data into the PNA's Active Channel
segs.SetAllSegments segdata

```

Download Segment Table

```

Option Explicit

Dim app

Set app = CreateObject("AgilentPNA835x.Application")

```

```

Dim chan

Set chan = app.ActiveChannel

chan.sweepType = 4

Dim segs

Set segs = chan.Segments

Dim win

Set win = app.NAWindows(1)

win.ShowTable 2

Dim segData

segData = segs.GetAllSegments

' Get lower bound and upper bound on the data values per each segment

Dim segDataLB, segDataUB

segDataLB = LBound(segData,1)

segDataUB = UBound(segData,1)

' Get lower bound and upper bound corresponding to how many segments

Dim segArrayLB, segArrayUB

segArrayLB = LBound(segData,2)

segArrayUB = UBound(segData,2)

' If the VB LBound and UBound functions didn't generate an error
' before reaching this point, that implies a valid two-dimensional
' array was returned into 'segData'.

WScript.Echo "Number of segments = " & segArrayUB - segArrayLB + 1

WScript.Echo "Number of data values per segment = " & segDataUB - segDataLB + 1

Dim index

index = segDataLB

Dim segInfStr

segInfStr = "Segment 1: state = " & segData(index, segArrayLB)

index = index + 1

```

```

segInfStr = segInfStr & ", num points = " & segData(index, segArrayLB)

index = index + 1

segInfStr = segInfStr & ", start freq = " & segData(index, segArrayLB)

index = index + 1

segInfStr = segInfStr & ", stop freq = " & segData(index, segArrayLB)

index = index + 1

segInfStr = segInfStr & ", IFBW = " & segData(index, segArrayLB)

index = index + 1

segInfStr = segInfStr & ", dwell time = " & segData(index, segArrayLB)

' In case of a measurement receiver PNA like N5264A
' which has no source ports, chan.SourcePortNames will
' return an empty variant (no array)

Dim srcPortNames

srcPortNames = chan.SourcePortNames

Dim srcPortNamesLB, srcPortNamesUB

srcPortNamesUB = -1

On Error Resume Next

srcPortNamesLB = LBound(srcPortNames)

srcPortNamesUB = UBound(srcPortNames)

On Error GoTo 0

If (srcPortNamesUB >= 0) And ((srcPortNamesUB - srcPortNamesLB + 1) <> (segDataUB
- index)) Then

    WScript.Echo "Mismatch in number of source port names!"

End If

Dim j

For j = index + 1 To segDataUB

    segInfStr = segInfStr & ", " & srcPortNames(j - (index + 1) + srcPortNamesLB)
    & " power = " & segData(j, segArrayLB)

Next

```

Last Modified:

28-Apr-2009 Added Download example

Create and Cal an SMC Measurement

This example creates and calibrates an SMC measurement. A power sensor must first be connected to the PNA.

By removing the comments (') at the start of the **BLUE code**, it can also do the following:

- Load a Mixer setup file from the PNA at: C:/Program Files/Agilent/Network Analyzer/Documents/Mixer/MyMixer.mxr.
- Cal using an ECal module.
- Perform manual ECAL orientation
- Load a [Phase Reference calibration](#).

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save on the PNA hard drive as SMC.vbs. Learn how to setup and run the macro.

See Also

[Create an SMC Fixed Output Meas](#)

[Use Existing Power Cal for SMC](#)

```
Dim App
Set App = CreateObject("AgilentPNA835x.Application")
App.Preset
Dim Meas
Set Meas = App.ActiveMeasurement
Meas.Delete
App.CreateCustomMeasurementEx 1, "Scalar Mixer/Converter","SC21"
'Other valid strings that can be specified to create a measurement with a
parameter
'other than "SC21" are: "S11", "S22", "IPwr", and "OPwr"
dim chan
set chan = app.activechannel
'Attenuator setting must match optional Phase Ref Cal setting
```

```
chan.attenuator(1) = 10

chan.NumberOfPoints = 11

chan.IFBandwidth = 1000

'You can perform mixer setup here or
'recall a previous mixer setup from the PNA Hard drive.
'This is how to perform mixer setup using IConverter.
' Setup Stimulus

dim cv

set cv = chan.Converter

cv.InputStartFrequency = 3.6e9

cv.InputStopFrequency = 4.9e9

cv.LOFixedFrequency(1) = 1e9

cv.LOPower(1) = 10

cv.OutputSideband = 0 'Lowside

cv.Calculate 2 'Calc output

'cv.EnablePhase = True

cv.LOName(1)="Port 3"

cv.Apply()

' Alternatively, recall a mixer setup from the PNA Hard drive

'Meas.LoadFile "C:/Program Files/Agilent/Network
Analyzer/Documents/Mixer/MyMixer.mxr"

' Begin Calibration

Dim CalMgr

Set CalMgr = App.GetCalManager

Dim SMC

Set SMC = CalMgr.CreateCustomCal("SMC")
```

```

SMC.Initialize 1, 1

SMC.ConnectorType(1) = "APC 3.5 male"
SMC.ConnectorType(2) = "APC 3.5 female"

' Use Mechanical cal kits

SMC.CalKitType(1) = "85033D/E"
SMC.CalKitType(2) = "85033D/E"

' To use an ECal module instead, comment out the above two lines
' and uncomment the appropriate lines below:
' Your ECal module must already be connected
' via USB to the PNA.
' SMC.CalKitType(1) = "N4691-60004 ECal"
' SMC.CalKitType(2) = "N4691-60004 ECal"
' Non-factory characterizations are specified as follows:
' SMC.CalKitType(1) = "N4691-60004 User 1 ECal"
' When two or more ECal modules with the same model number are connected
' also specify the serial number as follows:
' SMC.CalKitType(1) = "N4691-60004 ECal 01234"
' When Disk Memory ECal user characterizations are used,
' specify both the User char and the serial number as follows:
' SMC.CalKitType(1) = "N4691-60004 MyDskChar ECal 01234"
' Turn on auto orientation for the ECal (default behavior).
' SMC.AutoOrient = 1'

MsgBox("Cal kits defined for Ports 1 and 2")

' Import power cal data from an existing SMC calset.
' This calset MUST exist on the PNA.

SMC.ImportDataSet "Phase Reference-full-span","POWER_AND_PHASE"

'Omit the isolation part of the 2-port cal (default behavior).

SMC.OmitIsolation = 1

```

```
Dim steps

steps = SMC.GenerateSteps

For i = 1 To steps

    MsgBox SMC.GetStepDescription(i)

    SMC.AcquireStep i

Next

Dim calset

calset = SMC.GenerateErrorTerms

Msgbox("SMC Cal Complete!")
```

Last Modified:

8-Mar-2012	Edit for LOName (ZW)
24-Feb-2012	Added Phase Ref option
5-Apr-2011	Edited for ECal options
8-Mar-2011	Updated for A.09.33

Create and Cal a VMC Measurement

The following example program sets up a 1-stage mixer, then performs a VMC calibration using an N4691-60004 ECal module.

By removing the comments (') at the start of the **BLUE code**, it can also do the following:

- Use a mechanical cal kit
- Perform manual ECAL orientation
- Load a Mixer Characterization file

See Also

[Converter Object](#)

[VMCType Object](#)

[Example - Perform a VMC Mixer Characterization ONLY](#)

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as VMC.vbs. [Learn how to setup and run the macro.](#)

```
dim NASWEPT: NASWEPT = 0

dim NAFIXED: NAFIXED = 1

dim LOWSIDE: LOWSIDE = 0

dim HIGHSIDE: HIGHSIDE = 1

dim MIXEROUT: MIXEROUT = 2

dim pna: set pna = CreateObject("AgilentPNA835x.application")

pna.reset

' Create a VMC channel
' Other valid measurement strings are: "S11", and "S22"

pna.CreateCustomMeasurementEx 1, "Vector Mixer/Converter","VC21",1

' Setup Stimulus
```

```

dim chan: set chan = pna.activechannel

dim cv: set cv = chan.Converter

chan.NumberOfPoints = 11

chan.IFBandwidth = 1000

cv.InputStartFrequency = 3.6e9

cv.InputStopFrequency = 3.9e9

cv.LOFixedFrequency(1) = 1e9

cv.LOPower(1) = 10

cv.OutputSideband = LOWSIDE

cv.Calculate MIXEROUT

cv.LOName(1) = "Port 3"

cv.Apply()

DoBasicVMCCal (chan.channelNumber)

sub DoBasicVMCCal( channel )

dim myMixerCharFile: myMixerCharFile = "C:/Program Files/Agilent/Network
Analyzer/Documents/MyMixers2P.s2p"

' construct a VMC calibration object

dim calmanager: set calmanager = pna.GetCalManager

dim guidedCal: set guidedCal = calmanager.CreateCustomCalEx( channel )

dim vmc: set vmc = guidedCal.CustomCalConfiguration

' Initialize the cal object.

' Choose to respect or ignore the Preference: Cal: Auto Save to User Calset
' if you set this true, the behavior will be dependent on the setting
' of the preference.

dim useCalSetPreference: useCalSetPreference = false

```

```
vmc.Initialize channel, useCalSetPreference

' Define the DUT connectors and kits at ports 1 and 2 of the PNA
vmc.ConnectorType (1) = "APC 3.5 female"
vmc.ConnectorType (2) = "APC 3.5 male"

' Use Mechanical cal kits
vmc.CalKitType(1) = "85033D/E"
vmc.CalKitType(2) = "85033D/E"

' To use an ECal module instead, comment out the above two lines
' and uncomment the appropriate lines below:
' Your ECal module must already be connected
' via USB to the PNA.
' vmc.CalKitType(1) = "N4691-60004 ECal"
' vmc.CalKitType(2) = "N4691-60004 ECal"

' Non-factory characterizations are specified as follows:
' vmc.CalKitType(1) = "N4691-60004 User 1 ECal"

' When two or more ECal modules with the same model number are connected
' also specify the serial number as follows:
' vmc.CalKitType(1) = "N4691-60004 ECal 01234"

' When Disk Memory ECal user characterizations are used,
' specify both the User char and the serial number as follows:
' vmc.CalKitType(1) = "N4691-60004 MyDskChar ECal 01234"
'

MsgBox("Cal kits defined for Ports 1 and 2")

' By default, VMC requires the measurement of a Calibration Mixer.
' To determine the conversion loss of the calmixer, the cal wizard
' will add a step to perform a 1 port cal at the output of the mixer.
```

```
' The following commands opt to perform the mixer
' characterization using a cal kit.
' Do both characterization and full 2-port cal
vmc.CharacterizeMixerOnly = False
' Define the DUT connectors for the output of the characterization mixer
' Use (logical) Port 3. If it is already used by the DUT,
' then specify port 4.
vmc.ConnectorType(3) = "APC 3.5 male"
' Specify the mechanical cal kit for port 3
vmc.CalKitType(3) = "85033D/E"

' To avoid performing the 1-port cal steps, provide the wizard with a
' mixer characterization file. Uncomment the following two lines to
' specify the characterization file. This is an .S2P file.
' vmc.CharFileName = myMixerCharFile ' this file will be read
' vmc.LoadCharFromFile = true

' By default, auto orientation of the ecal module is performed
' Uncomment the following lines to manually orient the ecal
' vmc.autoorient = false
' for 2-port portion, ecal port A connected to PNA port 1
' vmc.EcalOrientation2Port(1) ="A1,B2"
' for mixer char, ecal port A connected to cal mixer output
' vmc.EcalOrientation1Port(1) = "A1"

' the main calibration loop
' a description for the connection instructions is read
' and then the standard is acquired
```

```
dim steps, connectionPrompt
steps = vmc.GenerateSteps
wscript.echo "Number of Steps = " + cstr(steps)
if (steps > 0) then ' otherwise an error condition occurred
for i = 1 to steps
    connectionPrompt = vmc.GetStepDescription( i )
    wscript.echo connectionPrompt
    vmc.AcquireStep( i )
next
vmc.GenerateErrorTerms
end if
end sub
```

Last Modified:

19-Feb-2013	Removed Mixer Char
1-Dec-2011	Added char mixer connector and cal kit.
5-Apr-2011	Updated for ECal options
16-Feb-2011	Modifications for FCA2

Create an SMC Fixed Output Measurement with COM

This VBScript example creates a calibrated SMC fixed output measurement using a controlled LO. Then a single sweep is taken and data is retrieved.

Requirements:

- If an external LO is used, it should be configured to match the LOName property of the mixer object.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as SMC.vbs. [Learn how to setup and run the macro.](#)

```
option explicit

' Utility function

function ToString(complexDataArray)

    dim dataAsString

    dim point

    for point = 0 to UBound(data)

        dataAsString = dataAsString & "(" & data(point,0) & "," & data(point,1) & " ) "

    next

    ToString = dataAsString

end function

dim app

set app = createobject("agilentpna835x.application")

app.preset

' Put the channel in hold (highly recommended)

app.ActiveChannel.Hold 1

' Delete the standard measurement

app.ActiveMeasurement.Delete
```

```

' Create an SC21 measurement
app.CreateCustomMeasurementEx 1, "Scalar Mixer/Converter","SC21"

' Set the number of points to 11
app.ActiveChannel.NumberOfPoints = 11

' Setup the mixer parameters for a swept LO, fixed output measurement
dim mixer

set mixer = app.ActiveMeasurement

mixer.InputStartFrequency = 200e6
mixer.InputStopFrequency = 700e6
mixer.LORangeMode(1) = 0 ' 0 = Swept mode
mixer.OutputFixedFrequency = 3.4e9
mixer.InputPower = -17
mixer.LOPower(1) = 10
'mixer.LOName(1) = "8360"

' The CALCULATE method calculates the LO frequency from the other parameters,
' It also applies ALL mixer parameters to the channel.
mixer.Calculate 3 ' Calculate the LO range

' Create an S11 in the same channel
app.CreateCustomMeasurementEx 1, "Scalar Mixer/Converter","S11"

dim S11Meas

set S11Meas = app.ActiveMeasurement

' Create an IPwr in the same channel
app.CreateCustomMeasurementEx 1, "Scalar Mixer/Converter","IPwr"

' Create an OPwr in the same channel
app.CreateCustomMeasurementEx 1, "Scalar Mixer/Converter","OPwr"

' Perform a single sweep synchronously.
app.ActiveChannel.Single 1

' Retrieve the SC21 data

```

```
dim data

'Get the calibrated values in polar format

data = mixer.GetData(1,3) ' 1 = naCorrectedData, 3 = naDataFormat_Polar

wscript.echo "SC21=" & ToString(data)

' Retrieve the S11 data

'Get the calibrated values in polar format

data = S11Meas.GetData(1,3) ' 3 = naDataFormat_Polar

wscript.echo "S11=" & ToString(data)
```

Last Modified:

8-Mar-2011	Updated for A.09.33
13-June-2006	Last updated

Create a Segmented Sweep for Mixers

This example program shows how to setup a segment sweep in FCA.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the PNA hard drive as Seg.vbs.

[Learn how to setup and run the macro.](#)

See Also

[Converter Object](#)

See Other COM Example Programs

```
option explicit

Dim app,chan,conv

Set app = CreateObject("AgilentPNA835x.Application")

Set chan = app.ActiveChannel

Set conv = chan.GetConverter

app.Reset

' Create FCA Scalar Mixer/Converter channel with an SC21 measurement:
app.CreateCustomMeasurementEx 1, "Scalar Mixer/Converter", "SC21", 1

' Delete all existing segments, and create three new ones

conv.DeleteAllSegments()

conv.AddSegment 1,3

' Turn on segment 1

conv.SegmentState(1)=True

' Set segment sweep

' The sweeptype command discards the changes made to the scratch mixer

' Therefore, precede with Apply

' Also, always do this before setting the LO port
```

```
conv.Apply

chan.SweepType = 4 'segment sweep

' Setup segment #1

' Input is swept from 1.1GHz to 1.39GHz

conv.SegmentStartFrequency(1,0)=1.1e9

conv.SegmentStopFrequency(1,0)=1.39e9

'Swept input

conv.SegmentRangeMode(1,0)=0

' Input power is -10 dBm

conv.SegmentFixedPower(1,0)=-10.0

' LO1 is fixed: 2.2 GHz

conv.SegmentFixedFrequency(1,2)=2.2e9

' LO1 power is 10.0 dBm

conv.SegmentFixedPower(1,2)=10.0

' Number of points is 21

conv.SegmentPoints(1)=21

' Output is swept

conv.SegmentRangeMode(1,1)=0

' Output is low-side

conv.SegmentMixingMode(1,1)=0

' Output is calculated from input and lo1

conv.SegmentCalculate 1,2

' Turn on segment 1

conv.SegmentState(1)=True

' Setup segment #2 from 1.40 to 1.49 GHz

' All else the same

conv.SegmentStartFrequency(2,0)=1.4e9

conv.SegmentStopFrequency(2,0)=1.49e9
```

```
conv.SegmentRangeMode(2,0)=0
conv.SegmentFixedPower(2,0)=-10.0
conv.SegmentFixedFrequency(2,2)=2.2e9
conv.SegmentFixedPower(2,2)=10.0
conv.SegmentPoints(2)=21
conv.SegmentRangeMode(2,1)=0
conv.SegmentMixingMode(2,1)=0
conv.SegmentCalculate 2,2
conv.SegmentState(2)=True
' Setup segment #3 from 1.50 to 1.59 GHz
' All else the same
conv.SegmentStartFrequency(3,0)=1.5e9
conv.SegmentStopFrequency(3,0)=1.59e9
conv.SegmentRangeMode(3,0)=0
conv.SegmentFixedPower(3,0)=-10.0
conv.SegmentFixedFrequency(3,2)=2.2e9
conv.SegmentFixedPower(3,2)=10.0
conv.SegmentPoints(3)=21
conv.SegmentRangeMode(3,1)=0
conv.SegmentMixingMode(3,1)=0
conv.SegmentCalculate 3,2
conv.SegmentState(3)=True
' Mixer Input to be port 1
' Mixer output to Port 2
' Mixer LO to Port 3
conv.LOName(1)="Port 3"
' Apply the scratch mixer
conv.Apply
```

Last Modified:

6-Jan-2012	Minor fixes and tested
16-Feb-2011	MX New topic

Use an Existing Power Cal During an SMC Cal

This example shows how to use an existing Source Power Cal instead of the power cal that is performed during an SMC calibration. To run this program without modification, you need the following:

- A Mixer setup file saved on the PNA: C:/Program Files/Agilent/Network Analyzer/Documents/Mixer/MyMixer.mxr.
- If the mixer file uses an external LO source, it must be connected and configured.
- An ECal module that covers the frequency range of the measurement.
- An SMC cal set named "SMC_CAL". This is the cal set that source power correction data will be imported from. The input and output frequency ranges of the cal set must cover the corresponding ranges used during calibration, or guided cal initialization will fail.

Error Messages

- If you attempt to import power cal data from an SMC calset that uses different ports than the ones currently in use, the message **"The necessary calibration standards were not found."** will appear.
- If the imported Cal Set does not cover the frequency range of the current cal, the message **"Interpolation target is out of range. Cannot interpolate."** will appear.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save on the PNA hard drive as SMC.vbs. Learn how to setup and run the macro.

See Also

[SMC Type Object](#)

[ImportDataSet Method](#)

See Other COM Example Programs

```
Dim App

Set App = CreateObject("AgilentPNA835x.Application")

App.Preset

Dim Meas

Set Meas = App.ActiveMeasurement
```

```

Meas.Delete

App.CreateCustomMeasurementEx 1, "Scalar Mixer/Converter","SC21"

'Other valid strings that can be specified to create a measurement with a
parameter

'other than"SC21" are: "S11", "S22", "IPwr", and "OPwr"

Set Meas = App.ActiveMeasurement

'You can perform mixer setup here or

'recall a previous mixer setup from the PNA Hard drive.

' This is how the mixer could be configured through the IMixer interface

Dim mix

Set mix = Meas ' reference to IMixer object

mix.ActiveXAxisRange = 0 ' 0 = mixINPUT (Input frequency range)

' Alternatively, recall a previous mixer setup from the PNA Hard drive

Meas.LoadFile "c:\users\public\network analyzer\documents\Mixer/MyMixer.mxr"

app.activechannel.numberofpoints = 21

Dim CalMgr

Set CalMgr = App.GetCalManager

Dim SMC

Set SMC = CalMgr.CreateCustomCal("SMC")

SMC.Initialize 1, 1

SMC.ConnectorType(1) = "APC 3.5 male"

SMC.ConnectorType(2) = "APC 3.5 female"

SMC.CalKitType(1) = "N4691-60004 ECal"

SMC.CalKitType(2) = "N4691-60004 ECal"

' Import power cal data from an existing SMC calset.

SMC.ImportDataSet "SMC_CAL","POWER_STEP"

'Omit the isolation part of the 2-port cal (default behavior).

SMC.OmitIsolation = 1

```

```
'Turn on auto orientation for the ECal (default behavior).
```

```
SMC.AutoOrient = 1
```

```
Dim steps
```

```
steps = SMC.GenerateSteps
```

```
For i = 1 To steps
```

```
    MsgBox SMC.GetStepDescription(i)
```

```
    SMC.AcquireStep i
```

```
Next
```

```
Dim calset
```

```
calset = SMC.GenerateErrorTerms
```

```
Msgbox("SMC Cal Complete!")
```

Last Modified:

8-Mar-2011 Updated for A.09.33

9-Feb-2011 Added error msg notes

Create a Balanced Measurement using COM

The following program creates several Balanced measurements in separate windows, generates markers, calculates statistics, and sets limit lines and queries results.

Note: By their nature, balanced measurements are extremely sensitive to phase differences between the two RF paths that make up the balanced port, especially at higher frequencies. A good calibration (not performed in this example) is critical to achieving good balanced measurement results.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as BalancedCOM.vbs. [Learn how to setup and run the macro.](#)

```
' PNA application object
Dim app

' Channel 1 object
Dim chan1

' start of marker/limit testing range
Dim minTestStimulus

' end of marker/limit testing range
Dim maxTestStimulus

' Set to true if you want additional balanced measurements.
Dim AdditionalMeasurements
AdditionalMeasurements = 1

' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
' Preset the instrument
app.Preset
' Get the Channel 1 object
Set chan1 = app.Channels(1)
' Stop data taking for now.
chan1.Hold true
' Set up the start / stop frequency for Channel 1 sweep.
MHZ = 1000000
GHZ = 1000*MHZ
chan1.StartFrequency = 10 *MHZ
chan1.StopFrequency = 1 *GHZ
chan1.NumberOfPoints = 801
' Define our test frequency range
minTestStimulus = 100*MHZ
maxTestStimulus = 900*MHZ
' This example uses DUT topology Bal-Bal -
```



```

app.CreateMeasurement 1, "BBAL:SDC11",1,2 ' C to D mode conversion reflection
app.CreateMeasurement 1, "BBAL:SCD11",1,2 ' D to C mode conversion reflection
app.CreateMeasurement 1, "BBAL:SCC11",1,2 ' common mode reflection

' Create Reverse Transmission Measurements, channel 1, window 3
app.CreateMeasurement 1, "BBAL:SDD12",1,3 ' differential mode transmission
app.CreateMeasurement 1, "BBAL:SCD12",1,3 ' differential to common mode conversion
app.CreateMeasurement 1, "BBAL:SDC12",1,3 ' common to differential mode conversion
app.CreateMeasurement 1, "BBAL:SCC12",1,3 ' common mode transmission

' Create (logical) Port 2 reflection measurements in window 4
app.CreateMeasurement 1, "BBAL:SDD22",1,4 ' differential mode reflection
app.CreateMeasurement 1, "BBAL:SDC22",1,4 ' C to D mode conversion reflection
app.CreateMeasurement 1, "BBAL:SCD22",1,4 ' D to C mode conversion reflection
app.CreateMeasurement 1, "BBAL:SCC22",1,4 ' common mode reflection
End If

' Set up some limit lines to verify a minimum differential insertion loss
sdd21_1.LimitTest(1).BeginStimulus = minTestStimulus
sdd21_1.LimitTest(1).EndStimulus   = maxTestStimulus
sdd21_1.LimitTest(1).BeginResponse = -2
sdd21_1.LimitTest(1).EndResponse   = -2
sdd21_1.LimitTest(1).Type = 2 ' minimum limit
sdd21_1.LimitTest.State = 1

' Limit lines for maximum common mode to differential conversion
sdc21_1.LimitTest(1).BeginStimulus = minTestStimulus
sdc21_1.LimitTest(1).EndStimulus   = maxTestStimulus
sdc21_1.LimitTest(1).BeginResponse = -20
sdc21_1.LimitTest(1).EndResponse   = -20
sdc21_1.LimitTest(1).Type = 1 ' maximum limit
sdc21_1.LimitTest.State = 1

' Take a (synchronous) single sweep on channel 1
chan1.Single true

' Show differential forward transmission statistics.
sdd21_1.ShowStatistics = true

' Set up user range 1 to limit marker's search range.
chan1.UserRangeMin(0,1) = minTestStimulus
chan1.UserRangeMax(0,1) = maxTestStimulus

' Find/Show max common mode to differential conversion, and read back the frequency.
sdc21_1.MarkerState(1) = true
' Set marker 1 to use user range 1
sdc21_1.Marker(1).UserRange = 1
sdc21_1.Marker(1).SearchMax

' Find/Show max differential mode insertion loss, and read back the frequency.
sdd21_1.MarkerState(1) = true
' Set marker 1 to use user range 1
sdd21_1.Marker(1).UserRange = 1
sdd21_1.Marker(1).SearchMin

```

```
If sdd21_1.LimitTestFailed Then
Wscript.Echo "Differential insertion loss failed: " & sdd21_1.Marker(1).Stimulus
/MHZ & "MHz, " & _sdd21_1.Marker(1).Value(1) & " dB"
End If
```

```
If sdc21_1.LimitTestFailed Then
Wscript.Echo "Common to differential conversion failed: " &
sdc21_1.Marker(1).Stimulus/MHZ & "MHz, " & _sdc21_1.Marker(1).Value(1) & " dB"
End If
```

Create a PMAR Device and Measurement

The following program creates a new External Device: a Power Meter as Receiver, makes several power meter settings, and then creates a PMAR measurement.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as PMAR.vbs. [Learn how to setup and run the macro.](#)

See Also

[ExternalDevices Collection](#)

[ExternalDevice Object](#)

[PowerSensorAsReceiver Object](#)

[PowerSensorCalFactorSegmentPMAR Object](#)

[PowerLossSegmentsPMAR_Collection](#)

[PowerLossSegmentPMAR Object](#)

```
dim app
Set app = CreateObject("AgilentPNA835x.Application")

dim externalDevices

Set externalDevices = app.ExternalDevices

dim devicecount

devicecount = externalDevices.count

externalDevices.Add "NewPMAR"

dim newExternalDevice

Set newExternalDevice = externalDevices.Item("NewPMAR")

newExternalDevice.DeviceType = "Power Meter"

newExternalDevice.IOConfiguration= "GPIB0::14::INSTR"

'newExternalDevice.IOConfiguration = "USB0::2391::4865::GB45100278::0::INSTR"

newExternalDevice.IOEnable = true
```

```
dim PMAR

Set PMAR = newExternalDevice.ExtendedProperties

PMAR.SensorIndex = 1

PMAR.ReadingsPerPoint = 10

dim avr

avr = PMAR.ReadingsPerPoint

PMAR.ReadingsTolerance = 0.1

dim tole

tole = PMAR.ReadingsTolerance

PMAR.MinimumFrequency = 100000000

PMAR.MaximumFrequency = 10000000000

PMAR.LimitFrequency = false

PMAR.referenceCalFactor = 99

Set powerCalFactorSegments = PMAR.CalFactorSegments

powerCalFactorSegments.Add 1,10

Set calpair = powerCalFactorSegments(1)

calpair.Frequency = 1e9

calpair.CalFactor = 99

Set calpair = powerCalFactorSegments(2)

calpair.Frequency = 2e9

calpair.CalFactor = 98

powerCalFactorSegments.Remove 3,8

PMAR.UsePowerLossSegments = true

Set pls = PMAR.PowerLossSegments

pls.Add 1,5

Set pl = pls(1)

pl.Loss = -1

pl.Frequency = 1e9
```

```
Set pl = pls(2)
pl.Loss = -2
pl.Frequency = 2e9
pls.Remove 3,3
newExternalDevice.active = true
'Create a PMAR trace with power meter connected to port 3
app.CreateMeasurement 1,"NewPMAR",3,1
```

Last Modified:

3-Sep-2009 MX New topic

Create a Wideband Pulsed Measurement using the PNA-X

This Visual Basic COM example shows you how to configure the PNA-X internal pulse generators and modulators to make wideband pulsed measurements in **pulse profile** mode using the PNA-X.

[Visit the PNA website](#) where you can download a free Wideband Pulsed Application that performs this measurement on the PNA-X.

[See all COM Pulsed examples](#)

```
'Create an PNA Application instance

Dim pnaApp As New AgilentPNA835x.Application

'Create a PathConfiguration instance

Dim pathConf As AgilentPNA835x.PathConfiguration

'Create an PulseGenerator instance

Dim pulseGen As AgilentPNA835x.PulseGenerator

'Create a Channel instance

Dim myChan As AgilentPNA835x.Channel

'Preset PNA-X

pnaApp.Preset

'Assign current active channel to myChan object

Set myChan = pnaApp.ActiveChannel

'Let PNA-X work in CW mode because of doing pulse profile measurement

myChan.SweepType = naCWTimeSweep

'Set CW Freq to 4 GHz

myChan.CWFrequency = 4000000000#

'Set IF Bandwidth to 5 MHz to get the best time resolution

myChan.IFBandwidth = 5000000#

'Assign current active channel path configuration to pathConf object

Set pathConf = myChan.PathConfiguration
```

```
'Let PNA-X source work in ALC Open Loop mode

myChan.ALCLevelingMode(1) = naALCOpenLoop

'Make the Pulse1 as modulation pulse to generate Pulsed-RF signal
pathConf.Element("PulseModDrive").Value = "Pulse1"

'Enable pulse modulation at Source1Out1 path
pathConf.Element("Src1Out1PulseModEnable").Value = "Enable"

'Assign current active channel pulse generator to pulseGen object
Set pulseGen = myChan.PulseGenerator

'Internal pulse generator has five channels,
'default the channel 0 use as internal ADC trigger signal
'Enable channel 0 of internal pulse generator as trigger signal
pulseGen.State(0) = True

'Enable channel 1 of internal pulse generator as modulation signal
pulseGen.State(1) = True

'Set pulse period to 10 us
pulseGen.Period = 0.00001 '10 us

'Set pulse width of channel 0 to 1 us
pulseGen.Width(0) = 0.000001 ' 1 us

'Set pulse width of channel 1 to 5 us
pulseGen.Width(1) = 0.000005 '5 us
```

End Sub

Last Modified:

4-Jan-2008 Added point trigger note
2-Oct-2007 MX New topic

Create an IM Spectrum Measurement

This VBScript example creates IM Spectrum measurement based on passed parameters.

This subprogram is extracted from the macro on the PNA that produces an IM spectrum channel from the Marker function. You can see the entire program at C:/Program Files/Agilent/Network Analyzer/Applications/IMD/IMD.VBS”.

This VBScript (*.vbs) program must be used as part of a program that supplies the required parameters. When complete, it can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as IMD.vbs.

[Learn how to setup and run the macro.](#)

[See SweptIMD Object.](#)

```
' ' SetupIMSpectrum
' ' Setup an IM Spectrum (non-converter) channel based upon supplied parameters
sub SetupIMSpectrum(app, MkrPos, ToneSpacing, TonePower)
    dim objIMXChan, objIMDChan
    dim Fstart, Fstop, NumPoints, ToneFc

    set objIMXChan = objIMSChan.CustomChannelConfiguration
    set objIMDChan = objSIMDChan.CustomChannelConfiguration

    NumPoints = objSIMDChan.NumberOfPoints
    select case objIMDChan.SweepType

    case naIMDToneCWSweep
        ToneFc = objIMDChan.FrequencyCenter

    case naIMDTonePowerSweep
        Fstart = objIMDChan.TonePowerStart(0)
        Fstop = objIMDChan.TonePowerStop(0)
```

```

    TonePower = CalcMkrValue(Fstart, Fstop, MkrPos, NumPoints)

    ToneFc = objIMDChan.FrequencyCenter

case naIMDToneCenterFreqSweep

    Fstart = objIMDChan.FrequencyCenterStart

    Fstop = objIMDChan.FrequencyCenterStop

    ToneFc = CalcMkrValue(Fstart, Fstop, MkrPos, NumPoints)

case naIMDDeltaFrequencySweep

    ToneFc = objIMDChan.FrequencyCenter

    Fstart = objIMDChan.DeltaFrequencyStart

    Fstop = objIMDChan.DeltaFrequencyStop

    ToneSpacing = CalcMkrValue(Fstart, Fstop, MkrPos, NumPoints)

case naIMDToneSegmentSweep

    ToneFc = MarkerXValue

end Select

objIMXChan.FrequencyCenter = ToneFc

objIMXChan.DeltaFrequency = ToneSpacing

objIMXChan.TonePower(0) = TonePower

objIMXChan.TonePower(1) = TonePower

app.ActiveMeasurement.Trace.ReferenceValue = TonePower + 10

objIMSChan.continuous

end Sub

```

Last Modified:

31-Mar-2009 MX New topic

Create an iTMSA Measurement

The following VB Script example shows how to create an iTMSA measurement with Power Sweep. Click each link to see a detailed description of each command.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save on the PNA hard drive as Testset.vbs. Learn how to setup and run the macro.

```
dim app      'PNA App
dim meas     'Measurement
dim balancemeas
dim balstimulus
dim chan

set app = createobject("agilentpna835x.application")
set chan = app.activechannel
chan.SweepType = 2 ' Set the sweep type to power sweep

set meas = app.ActiveMeasurement
set balancemeas = meas.BalancedMeasurement
balancemeas.BalancedTopology.DUTTopology = 2 ' Bal-Bal topology
balancemeas.BalancedStimulus.Mode = 1 ' Turn on true mode
'The PNA-X balanced port numbers are always (0)=Bal 1; (-1)=Bal2
chan.StartPowerEx(0) = -5 ' Set the balanced port 1 start power to -5 dbm
chan.StopPowerEx(0) = 5 ' Set the balanced port 1 stop power to 5 dbm
chan.StartPowerEx(-1) = -10 ' Set the balanced port 2 start power to -5 dbm
chan.StopPowerEx(-1) = 0 ' Set the balanced port 2 stop power to 5 dbm
```

Last Modified:

2-Jun-2008 MX New topic

Create and Cal a Gain Compression Measurement

This VBScript example creates and calibrates a Gain Compression measurement and performs Compression analysis.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as GCA.vbs. Learn how to setup and run the macro .

See Gain Compression Object.

`option explicit`

```
dim CompLevel , Tolerance , StartFreq , StopFreq , NumFreqs , Scale ,  
LinearPower
```

```
dim AcqMode , BackOff , StartPower , StopPower , NumPowers ,  
EnableInterp , CompAlg
```

```
dim DwellTime , IFBandwidth , ShowIterations , host , app
```

```
' GCA Settings/Values
```

```
''
```

```
' Acquisition Mode:
```

```
' naSmartSweep = 0
```

```
' naSweepPowerAtEachFreq2D = 1
```

```
' naSweepFreqAtEachPower2D = 2
```

```
''
```

```
' Compression Algorithm
```

```
' naCompressionFromLinearGain = 0
```

```
' naCompressionFromMaximumGain = 1
```

```
' naBackoffCompression = 2
```

```
' naXYCompression = 3
```

```
''
```

```
' ' EndOfSweepOperation
' ' naDefaultPowerSet = 0
' ' naSetToStartPower = 1
' ' naSetToStopPower = 2
' ' naSetRFOff = 3
''

CompLevel          = 1          ' 1 dB compression level
Tolerance          = 0.05      ' SMART Sweep tolerance
StartFreq          = 1E9
StopFreq           = 9E9
NumFreqs           = 201
Scale              = 0.1
LinearPower        = -20
BackOff            = 10        ' Not used for Deviation from linear gain
StartPower         = -20
StopPower          = 8
NumPowers          = 60        ' Not used for SMART Sweep
DwellTime          = 0.0005    ' Allow some time for DUT bias/thermal
effects
IFBandwidth        = 1000      ' Reasonable trace noise at -20 dBm
EnableInterp       = False     ' Disable interpolation
AcqMode            = 0          ' Smart Sweep
CompAlg            = 0          ' Deviation from linear gain
ShowIterations     = False     ' Configure SMART to not show iteration
results
```

```
dim objargs
set objargs = wscript .Arguments
if (objArgs .Count = 1) then host = objargs (0)

.....

'' Create and Configuration GCA Channel:
.....

set app = CreateObject ("Agilentpna835x.application" )
call SetupGCA ( app ,_
                StartFreq ,_
                StopFreq ,_
                NumFreqs ,_
                EnableInterp ,_
                Scale ,_
                CompLevel ,_
                LinearPower ,_
                AcqMode ,_
                BackOff ,_
                StartPower ,_
                StopPower ,_
                NumPowers ,_
                CompAlg ,_
                DwellTime ,_
                IFBAndwidth ,_
```

```

        ShowIterations )

call CalGCA ( app )

call Analysis ( app )

.....

'' GCA Setup
.....

sub SetupGCA ( app , StartFreq , StopFreq , NumFreqs , EnableInterp ,
Scale , CompLevel , LinearPower , _
, AcqMode , BackOff , StartPower , StopPower , NumPowers
, CompAlg , DwellTime , IFBAndwidth , _
        ShowIterations )

dim chan , gca
app .reset
app .CreateCustomMeasurementEx 1, "Gain Compression" , "S21" , 1
set chan = app .channels (1)
chan .hold 1

app .CreateCustomMeasurementEx 1, "Gain Compression" , "CompIn21" ,
1

app .CreateCustomMeasurementEx 1, "Gain Compression" , "DeltaGain21"
, 1

app .nawindows (1).traces (3).YScale = Scale
app .nawindows (1).traces (3).ReferenceValue = -CompLevel
set gca = chan .CustomChannelConfiguration
gca .InputLinearPowerLevel = LinearPower
gca .AcquisitionMode = AcqMode
gca .CompressionLevel = CompLevel
gca .CompressionBackoff = BackOff
gca .CompressionDeltaX = BackOff

```

```
gca .CompressionDeltaY = BackOff - CompLevel
gca .CompressionAlgorithm = CompAlg
gca .NumberOfPowerPoints = NumPowers
gca .CompressionInterpolation = EnableInterp
gca .SmartSweepSettlingTime = DwellTime
gca .SmartSweepShowIterations = ShowIterations
chan .IFBandwidth = IFBandwidth
chan .DwellTime = DwellTime
chan .StartPower = StartPower
chan .StopPower = StopPower
chan .StartFrequency = StartFreq
chan .StopFrequency = StopFreq
chan .NumberOfPoints = NumFreqs
chan .single 1
```

end sub

```
.....
'' GCA Calibration
.....
```

sub CalGCA (app)

```
dim chan , CalMgr , GCACal , GCACustomCal , CalSteps , I , CalSet
set chan = app .ActiveChannel
set CalMgr = app .GetCalManager
set GCACal = CalMgr .CreateCustomCalEx (1)
' Configure GCA Guided Cal for the connector types and ECal module
```

that will be used:

```
GCACal .Initialize 1, 0
GCACal .ConnectorType (1) = "APC 3.5 female"
GCACal .ConnectorType (2) = "APC 3.5 male"
GCACal .CalKitType (1) = "N4691-60004 ECal"
GCACal .CalKitType (2) = "N4691-60004 ECal"
set GCACustomCal = GCACal .CustomCalConfiguration
GCACustomCal .PowerLevel = 0      ' Set power level for source cal to
0 dBm
CalSteps = GCACal .GenerateSteps
for I = 1 to CalSteps
  msgBox GCACal .GetStepDescription (I )
  GCACal .AcquireStep (I )
next
CalSet = GCACal .GenerateErrorTerms ' Calculate error terms and
apply CalSet to GCA Channel
chan .CalSet .Save ("GCA 2P" ) ' Save CalSet
msgBox "Done"
end sub
```

```
.....
'' GCA Analysis (PNA Rev A.09.00 or later)
.....
```

```
sub Analysis (app)
Dim meass
Dim ana
Set meass = app.Measurements ' get the measurements
```

```
Set ana = meass(1).CustomMeasurementConfiguration 'get the
measurement
ana.AnalysisEnable = true ' enable the analysis mode
ana.AnalysisCWFreq = 3e9 ' set the analysis cw frequency to 3GHz
Set ana = meass(2).CustomMeasurementConfiguration
ana.AnalysisEnable = true
ana.AnalysisCWFreq = 4e9
ana.AnalysisXAxis = naPsourceAsXAxis ' set the XAxis as the source
power setting
Set ana = meass(3).CustomMeasurementConfiguration
ana.AnalysisEnable = true
ana.AnalysisIsDiscreteFreq = false ' turn off the discrete frequency
option
ana.AnalysisCWFreq = 4.5e9
end sub
```

Last Modified:

3-Sep-2009	Added GCA analysis
17-Dec-2007	MX New topic

Create and Cal a GCX Measurement

This VBScript example creates and calibrates a GCX measurement and performs Compression analysis.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as GCX.vbs. [Learn how to setup and run the macro.](#)

[See Gain Compression Object and Converter Object](#)

```
option explicit

dim CompLevel, Tolerance, StartFreq, StopFreq, LOFreq, NumFreqs, Scale,
LinearPower

dim AcqMode, BackOff, StartPower, StopPower, NumPowers, EnableInterp, CompAlg

dim DwellTime, IFBandwidth, ShowIterations, host, app, parser

CompLevel      = 1      ' 1 dB compression level
Tolerance       = 0.05  ' SMART Sweep tolerance
StartFreq      = 2.5E9
StopFreq       = 2.6E9
LOFreq = 1.7E9
NumFreqs       = 21
Scale          = 0.1
LinearPower    = -10
BackOff        = 10    ' Not used for Deviation from linear gain
StartPower     = -20
StopPower      = 8
NumPowers      = 60    ' Not used for SMART Sweep
DwellTime     = 0.0005 ' Allow some time for DUT bias/thermal effects
IFBandwidth    = 1000  ' Reasonable trace noise at -20 dBm
EnableInterp   = False ' Disable interpolation
AcqMode        = 0     ' Smart Sweep
```

```

CompAlg          = 0          ' Deviation from linear gain

ShowIterations   = False     ' Configure SMART to not show iteration results

dim objargs

set objargs = wscript.Arguments

if (objArgs.Count = 1) then host = objargs(0)

'-----

' Create and Configuration GCX Channel:

'-----

set app = CreateObject("Agilentpna835x.application")

call SetupGCAX( parser, _
                StartFreq, _
                StopFreq, _
                LOFreq, _
                NumFreqs, _
                EnableInterp, _
                Scale, _
                CompLevel, _
                LinearPower, _
                AcqMode, _
                BackOff, _
                StartPower, _
                StopPower, _
                NumPowers, _
                CompAlg, _
                DwellTime, _
                IFBAndwidth, _
                ShowIterations )

call CalGCAX( parser )

```

```

'-----
' GCAX Setup
'-----

sub SetupGCAX( parser, StartFreq, StopFreq, LOFreq, NumFreqs, EnableInterp,
Scale, CompLevel, LinearPower, _
                AcqMode, BackOff, StartPower, StopPower, NumPowers, CompAlg,
DwellTime, IFBAndwidth, _
                ShowIterations )

dim chan, gca

app.reset

app.CreateCustomMeasurementEx 1, "Gain Compression Converters", "SC21", 1

set chan = app.channels(1)

dim converter

set converter = chan.Converter()

chan.hold 1

app.CreateCustomMeasurementEx 1, "Gain Compression Converters", "CompIn21", 1
app.CreateCustomMeasurementEx 1, "Gain Compression Converters", "DeltaGain21", 1
app.nawindows(1).traces(3).YScale = Scale
app.nawindows(1).traces(3).ReferenceValue = -CompLevel

set gca = chan.CustomChannelConfiguration

gca.InputLinearPowerLevel = LinearPower

gca.AcquisitionMode = AcqMode

gca.CompressionLevel = CompLevel

gca.CompressionBackoff = BackOff

gca.CompressionDeltaX = BackOff

gca.CompressionDeltaY = BackOff - CompLevel

gca.CompressionAlgorithm = CompAlg

gca.NumberOfPowerPoints = NumPowers

gca.CompressionInterpolation = EnableInterp

```

```

gca.SmartSweepSettlingTime = DwellTime

gca.SmartSweepShowIterations = ShowIterations

chan.IFBandwidth = IFBandwidth

chan.DwellTime = DwellTime

chan.StartPower = StartPower

chan.StopPower = StopPower

chan.TestPortPower(1) = LinearPower

chan.StartFrequency = StartFreq

chan.StopFrequency = StopFreq

chan.NumberOfPoints = NumFreqs

'set converter properties

converter.InputRangeMode = 0 ' swept

converter.LORangeMode(1) = 1 'fixed

converter.OutputRangeMode = 0 'swept

converter.InputStartFrequency = StartFreq

converter.InputStopFrequency = StopFreq

converter.LOFixedFrequency(1) = LOFreq

converter.LOName(1) = "Port 3"

converter.LOPower(1) = -10

converter.Calculate 2 'calculateOutput

chan.Single 1

end sub

'-----

' GCAX Calibration

'-----

sub CalGCAX( parser )

Dim CalMgr

Set CalMgr = app.GetCalManager

```

```
Dim SMC

Set SMC = CalMgr.CreateCustomCal("SMC")

SMC.Initialize 1, 1

SMC.Do2PortEcal = 1 'specify 0 for mechanical cal, 1 for ecal

'use Factory Characterization
SMC.ECALCharacterization(1) = 0

SMC.OmitIsolation = 1

SMC.AutoOrient = 1

' 1- forward, 2-reverse, or Both

SMC.CalibrationPort = "1"

Dim steps

steps = SMC.GenerateSteps

Dim i

For i = 1 To steps

    MsgBox SMC.GetStepDescription(i)

    SMC.AcquireStep i

Next

Dim calset

calset = SMC.GenerateErrorTerms

Msgbox("SMC Cal Complete!")

end sub
```

Last Modified:

3-Mar-2011 Changed to converter property

Create and Cal a Noise Figure Measurement

This example program creates a Noise Figure measurement, then calibrates the measurement.

You MUST change the ECal Identification strings (in **Blue** font).

Optional: Uncomment the following lines (in **Blue** font) to change these settings:

- Noise Receiver = Noise Receiver to Std (PNA) Receiver
- Cal Method = "Vector" to "Scalar"
- Receiver Characterization Method = "NoiseSource" to "Power Meter"

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Noise.vbs. [Learn how to setup and run the macro.](#)

See Also

[NoiseFigure Object.](#)

[Accessing the NoiseFigure object and NoiseCal object using C#](#)

See other COM Examples

```
windowNum = 1

channelNum = 1

set pna=CreateObject("AgilentPNA835x.Application")

set scpi = pna.ScpiStringParser

pna.reset

' Create noise figure measurement

set noise = pna.createcustommeasurementex(channelNum, "NoiseFigure", "NF",
windowNum)

set noisechan = pna.activechannel

' Create object to access noise-specific channel attributes

set noiseConfig = pna.activechannel.CustomChannelConfiguration
```

```

' Create guided noise calibration object on our channel

set noisecal = pna.GetCalmanager.CreateCustomCalEx(channelNum)

set noiseCalExtension = noisecal.CustomCalConfiguration

noiseCalExtension.NoiseSourceCold = 300

' Substitute appropriate ECal identification strings here

tunerEcal = "N4691-60004 ECal 02821"

pullEcal = "N4691-60004 ECal 02297"

' configuration

ConfigureChannel

ConfigureNoiseSettings

' perform calibration

SetupCalAttributes_Insertable

SetupNoiseSource

FinishCalibration

' ----- Support subroutines -----

' Configure noise channel

sub ConfigureChannel

    noisechan.startfrequency = 500e6

    noisechan.stopfrequency = 5.0e9

    noisechan.numberofpoints = 201

    noisechan.IFBandwidth = 1.0E3

end sub

' Configure noise-specific channel settings

sub ConfigureNoiseSettings

    noiseConfig.NoiseReceiver = 1 'Noise Receiver

' noiseConfig.NoiseReceiver = 0 'Std PNA Receiver

    noiseConfig.noiseaveragestate = true

    noiseConfig.NoiseAverageFactor = 40

```

```

noiseConfig.NoiseTuner = tunerEcal

noiseConfig.NoiseTunerIn = "B"

noiseConfig.NoiseTunerOut = "A"

noiseConfig.NoiseBandwidth = 8e6

end sub

sub SetupCalAttributes_Insertable

    noiseCal.Initialize channelNum, true

    noiseCal.ConnectorType( 1 ) = "APC 3.5 female"

    noiseCal.ConnectorType( 2 ) = "APC 3.5 male"

    noiseCal.CalKitType (1) = pullEcal

    noiseCal.CalKitType (2) = pullEcal

    noiseCalExtension.NoiseSourceConnectorType = "APC 3.5 male"

    noiseCalExtension.NoiseSourceCalKitType = pullEcal

    noiseCalExtension.CalMethod = "Vector"

' noiseCalExtension.CalMethod = "Scalar"

    noiseCalExtension.RcvCharMethod = "NoiseSource" 'Can NOT be used with Std PNA
Rcvr

' noiseCalExtension.RcvCharMethod = "PowerMeter"

end sub

sub SetupNoiseSource

' specify the ENR file for the noise source

    noiseCalExtension.ENRFile = "c:\users\public\network
analyzer\documents\346C_MY44420454.enr"

    noiseCalExtension.NoiseSourceCold = 301.1

end sub

' Build the connection list and acquire the calibration

sub FinishCalibration

```

```

steps = noisecal.GenerateSteps

for i = 1 to steps

    str = noisecal.GetStepDescription( i )

    msgbox str

    noisecal.AcquireStep i

next

guid = noisecal.GenerateErrorTerms

wscript.echo "Calibration created calset guid: ",guid

end sub

```

Bonus: Accessing the NoiseFigure object and NoiseCal object using C#

Replace *<hostname>* with the full computer name of your PNA

```

Type pna = Type.GetTypeFromProgID("AgilentPNA835x.Application", "<hostname>");

    AgilentPNA835x.Application app =
(AgilentPNA835x.Application)Activator.CreateInstance(pna);

    app.Reset();

    app.CreateCustomMeasurementEx(1, "NoiseFigure", "NF", 1);

    AgilentPNA835x.ICalManager5 calManager =
(AgilentPNA835x.ICalManager5)app.GetCalManager();

    AgilentPNA835x.IGuidedCalibration4 guidedCal4 =
(AgilentPNA835x.IGuidedCalibration4)

        calManager.CreateCustomCalEx(1);

    AgilentPNA835x.INoiseCal noiseCal =
(AgilentPNA835x.INoiseCal)guidedCal4.CustomCalConfiguration;

```

Last Modified:

9-Jun-2011	Edited for Rcvr Char method (A.09.41)
14-May-2010	Fixed hostname
9-Nov-2007	MX New topic

Create and Cal an NFX Measurement

This program does the following:

- Setup a Noise Figure SC21 Measurement
- Calibrate Noise Figure channel
- Optional - Configure for an Embedded LO

To run this program, make the following edits, highlighted in **yellow**:

- Set **host** to your PNA computer name
- Set **tunerECal** and **pullECal** to your ECal model and info
- Set **ENR** to correct file name and location
- Set **connector types** for ECal, power sensor, and noise source

This program can be run as a macro in the PNA. To do this, copy the code into a text editor file such as Notepad and save on the PNA hard drive as NFX.vbs. [Learn how to setup and run the macro.](#)

See Also

[CreateCustomMeasEX command](#)

[NoiseFigure Object](#)

[NoiseCal Object](#)

[Converter_Object](#)

[GuidedCal Object](#)

[EmbeddedLO Object](#)

Learn About...

[Noise Figure on Converters](#)

[Noise Cal](#)

[See other COM Examples](#)

```
option explicit

'NFX sweep type
' naLinearSweep = 0
' naCWTimeSweep = 2

'Converter sweep mode
' naSwept = 0
' naFixed = 1

'Embedded LO tuning mode
' Broadband and precise = 0
' Precise only = 1
' None = 2

dim app
dim chan
dim nfx
dim host

host = "MyPNA"

set app = CreateObject("Agilentpna835x.application", host)

app.reset

call SetupNFX

'optional if not doing embedded LO
'call SetupEmbeddedLO

call CalNFX

sub SetupNFX
dim tunerEcal
tunerEcal = "N4691-60003 ECal 00591"

' create NFX traces
```

```
app.CreateCustomMeasurementEx 1, "Noise Figure Converters", "NF", 1
app.CreateCustomMeasurementEx 1, "Noise Figure Converters", "SC21", 1

'set channel and application objects

set chan = app.ActiveChannel

set nfx = chan.CustomChannelConfiguration

dim converter

set converter = chan.GetConverter()

' Set channel properties

chan.single 1

chan.sweepType = 0 'naLinearSweep

chan.numberOfPoints = 201

chan.IFBandwidth = 1.e3

' Set nfx properties

nfx.noiseAverageState = true

nfx.noiseAverageFactor = 10

nfx.noiseTuner = tunerECal

nfx.NoiseTunerIn = "B"

nfx.NoiseTunerOut = "A"

nfx.NoiseBandwidth = 8e6

nfx.noiseGain = 0 'low

' converter properties

converter.InputRangeMode = 0 ' swept

converter.LORangeMode(1) = 1 'fixed

converter.OutputRangeMode = 0 'swept

converter.InputStartFrequency = 8.0e8
```

```
converter.InputStopFrequency = 3.0e8
converter.LOFixedFrequency(1) = 1.5825e10
converter.LOPower(1) = -10
converter.Calculate 2 'calculateOutput
converter.LOName(1) = "Port 3"

converter.Apply

chan.Single 1

end sub

sub CalNFX
' Set ecal and noise tuner

dim SparamECal

SparamECal = "N4693-60001 User 2 ECal 00012"

chan.single 1

dim calMgr
set calMgr = app.GetCalManager

dim nfxCal
set nfxCal = CalMgr.CreateCustomCalEx(1)

dim nfxCalExt
set nfxCalExt = nfxCal.CustomCalConfiguration

nfxCalExt.ENRFile = "C:/Program Files/Agilent/Network
Analyzer/Noise/346C_44420601.enr"

'setup calibration

nfxCal.Initialize 1, true

'dut connector

nfxCal.ConnectorType(1) = "APC 3.5 female"
nfxCal.ConnectorType(2) = "APC 3.5 female"
```

```
nfxCal.CalKitType(1) = SparamECal
nfxCal.CalKitType(2) = SparamECal
'power sensor connector
nfxCal.PowerCalibrationPowerLevel(1) = -20
nfxCal.PowerSensorConnectorType(2) = "APC 3.5 male"
nfxCal.PowerSensorCalkitType(2) = SparamECal
'noise source connector
nfxCalExt.NoiseSourceConnectorType = "APC 3.5 male"
nfxCalExt.NoiseSourceCalKitType = SparamECal
nfxCalExt.CalMethod = "Vector"
nfxCalExt.EnableLOPowerCal(1) = False
nfxCalExt.ForceDeEmbedENRAdapter = False
nfxCalExt.ForceDeEmbedSensorAdapter = False

'step through calsteps
dim steps
steps = nfxcal.GenerateSteps
dim i , str
for i = 1 to steps
str = nfxcal.GetStepDescription(i)
msgbox str
nfxcal.AcquireStep i
next
dim guid
guid = nfxcal.generateerrorterms
wscript.echo "Calibration created calset guid: ", guid
chan.continuous
end sub
```

```
sub SetupEmbeddedLO
' Set embedded LO properties

dim ELO

set ELO = converter.ConverterEmbeddedLO

ELO.NormalizePoint = 101

ELO.TuningMode = 0 ' Broadband and precise

ELO.TuningIFBW = 3.0e4

ELO.MaxPreciseTuningIterations = 5

ELO.PreciseTuningTolerance = 1

ELO.TuningSweepInterval = 1

ELO.IsOn = true

chan.Single 1

end sub
```

Last Modified:

26-Oct-2009 MX New topic

Create and Cal a Swept IMD Measurement

This VBScript example creates IMD power and IM3 measurements, sets sweep mode to Center Frequency Sweep, and performs an IMD cal.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as IMD.vbs.

You can see the VB Script program on the PNA that produces an IM spectrum channel from the Marker function at :
C:/Program Files/Agilent/Network Analyzer/Applications/IMD/IMD.VBS".

[Learn how to setup and run the macro.](#)

[See SweptIMD Object.](#)

```
option explicit

'declare variables

dim SweepMode, StartDeltaFreq, StopDeltaFreq, NumFreqs, TonePower, CWFreq

dim app, hostname

'' Sweep type:

'' naIMDToneCWSweep          = 0
'' naIMDTonePowerSweep      = 1
'' naIMDToneCenterFreqSweep = 2
'' naIMDDeltaFrequencySweep = 3
'' naIMDToneSegmentSweep    = 4

'init variables

SweepMode    = 3          ' Sweep DeltaF

StartDeltaFreq  = 100e3

StopDeltaFreq  = 1e9

NumFreqs      = 201

TonePower     = -7

CWFreq        = 5e9

' get host name from commandline
```

```

dim objargs
set objargs = wscript.arguments
if(objargs.Count = 1) then hostname = objargs(0)
set app = CreateObject("Agilentpna835x.application", hostname)
call SetupIMD
call CalIMD
.....
'' Create and Configure IMD channel
.....

sub SetupIMD

    dim chan, imd

    app.reset

    ' Create IMD measurements

    app.CreateCustomMeasurementEx 1, "Swept IMD", "PwrMain", 1
    app.CreateCustomMeasurementEx 1, "Swept IMD", "IM3", 1

    set chan = app.channels(1)

    chan.hold 1

    set imd = chan.CustomChannelConfiguration

    imd.SweepType = SweepMode

    imd.FrequencyCenter = CWFreq

    imd.DeltaFrequencyStart = StartDeltaFreq
    imd.DeltaFrequencyStop  = StopDeltaFreq

    imd.TonePower(0)         = TonePower 'F1 power
    imd.TonePower(1)         = TonePower 'F2 power

    chan.NumberOfPoints = NumFreqs

```

```

chan.single 1

end sub

sub CalIMD

dim chan, CalMgr, IMDCal, IMDCustomCal, CalSteps, I, CalSet

set chan = app.ActiveChannel

set CalMgr = app.GetCalManager

set IMDCal = CalMgr.CreateCustomCalEx(1)

'Configure IMD GuidedCal for the connector types and ECal module that will be
used

' Substitute appropriate connector type and ECal identification strings here
IMDCal.Initialize 1, true 'channel number is 1
IMDCal.ConnectorType(1) = "APC 3.5 female"
IMDCal.ConnectorType(2) = "APC 3.5 male"
IMDCal.CalKitType(1) = "N4693-60001 User 2 ECal 00012"
IMDCal.CalKitType(2) = "N4693-60001 User 2 ECal 00012"

' IMD Custom settings

set IMDCustomCal = IMDCal.CustomCalConfiguration

' Set the Power Level at the power sensor to be used in calibration

IMDCustomCal.PowerLevel = 0

' Specify the connector type of the power sensor. If there is an adapter
between

' the input port and the power sensor, specify the connector type here, and
set

' the appropriate cal kit type for the connector so that extra calibration
can be

' performed. To skip the calibration for the adapter, set

```

```
PowerSensorConnectorType to "Ignored"

' i.e.: IMDCustomCal.PowerSensorConnectorType = "Ignored"

IMDCustomCal.PowerSensorConnectorType = "APC 3.5 female"

IMDCustomCal.PowerSensorCalKitType = "N4693-60001 User 2 ECal 00012"

' Set the Max product to calibrate, valid values are 3, 5, 7, and 9

IMDCustomCal.MaxProduct = 3

' Set the calibration Frequencies, can choose between calibrate only at
center Frequencies (0)

' or calibrate at all frequencies (1).

IMDCustomCal.CalibrationFrequencies = 1

'Include 2nd order product in calibration

IMDCustomCal.Include2ndOrderProduct = true

CalSteps = IMDCal.GenerateSteps

for I = 1 to CalSteps

    msgBox IMDCal.GetStepDescription(I)

    IMDCal.AcquireStep(I)

next

CalSet = IMDCal.GenerateErrorTerms

msgBox "IMD Cal Done"

end sub
```

Last Modified:

31-Mar-2009 MX New topic

ENR File Management Example

This VB Script program illustrates ENR file management using COM commands.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save on the PNA hard drive as Calibrate.vbs. Learn how to setup and run the macro.

[See Other COM Example Programs](#)

```
' Sample VBS program illustrating COM commands for ENR file management.

option explicit

dim pna    ' application
dim enr    ' ENRFile object
dim scpi, hostname

set pna=CreateObject("agilentpna835x.application")
set scpi = pna.ScpiStringParser
set enr = pna.ENRFile

' Generate data to put in ENR file
Dim vdata(3)
vdata(0) = 100E6    ' first frequency point
vdata(1) = 14.532   ' first ENR value
vdata(2) = 20E9     ' second frequency point
vdata(3) = 15.731   ' second ENR value

' send data to ENRFile object
enr.PutENRData(vdata)
```

```
' Set noise source serial number
```

```
enr.ENRSN = "ABCD1234"
```

```
' Write ENR file to disk
```

```
enr.SaveENRFile("C:/Program Files/Agilent/Network Analyzer/Documents/sample.enr")
```

The contents of the file created by this program are shown below.

[Filetype ENR]

[Version 1.0]

[Serialnumber ABCD1234]

! Frequency ENR

! Hz dB

100000000 14.53200

2e+010 15.73100

Last Modified:

14-May-2010 Fixed hostname

2-Aug-2007 MX New topic

Events with C++

The following code, along with the Header file, shows how to use the PNA Events.

[Download the Header file 'preventcatcher.h'](#)

```
#include <atlbase.h>

#include <atlcom.h>

#include <iostream>

#import "835x.tlb" no_namespace,raw_interfaces_only,named_guids

#include "pnaeventcatcher.h"

inline void HR(HRESULT hr)

{

    if (FAILED(hr))

        throw hr;

}

class MyEventCatcher : public CPNAEventCatcher

{

public:

    MyEventCatcher()
```

```

{
    CoInitialize(NULL);
    CComPtr<IApplication> app;
    HR(app.CoCreateInstance(CLSID_Application));
    CPNAEventCatcher::SubscribeCatcher(app);
    HR(app->AllowAllEvents());
}
~MyEventCatcher()
{
    CPNAEventCatcher::Release();
    CoUninitialize();
}
virtual void OnMeasurementEvent(long eventID,long measurementNumber) {}
virtual void OnChannelEvent(long eventID,long ch)
{
    if (eventID == 0x68070709L) //MSG_ALL_SWEEPS_COMPLETED_AND_PROCESSED
    {
        static int i =0;
        ++i;
        std::cout << "Sweep:" << i << std::endl;
    }
}
};

```

In a .cpp file, (just like most ATL projects) you must have a declared an instance of CComModule.
This will work:

```
CComModule _Module;
```

```
-----
```

Remember that you are now the "Server" and the PNA is the Client. That makes DCOM a bit complicated.

This code was tested in VS2005 using a wizard generated MFC MDI project.

Last Modified:

13-Nov-2007 MX New topic

FOM Examples

All three VBScript examples in this topic create a FOM measurement with the following attributes:

- Sweep the Source (input) from 1 GHz to 2 GHz
- Sweep the Receivers (output) from 2 GHz to 3 GHz
- You provide an LO at 1 GHz

[Learn more about Frequency Offset Mode](#)

These programs can be run as a macro in the PNA. To do this, copy the code into a text editor file such as Notepad and save on the PNA hard drive as FOM.vbs. [Learn how to setup and run the macro.](#)

The following example will run on any PNA model with FOM (opt 080). however, These commands have no provisions for internal second source. It uses commands introduced before 'enhanced FOM' was released for the A.07.10 release.

```
set app = CreateObject("Agilentpna835x.application")
set chan = app.ActiveChannel
chan.startFrequency = 1e9
chan.StopFrequency = 2e9
' set the receiver frequencies to be 2e9->3e9
chan.FrequencyOffsetFrequency = 1e9
chan.FrequencyOffsetState = 1
```

The following example can be run ONLY on a PNA with revision A.07.10 or later and has FOM (opt 080). It uses new FOM commands. [See FOMRange object.](#)

```
set app = CreateObject("Agilentpna835x.application")

set chan = app.ActiveChannel

chan.startFrequency = 1e9

chan.StopFrequency = 2e9

' set the receiver frequencies to be 2e9->3e9

chan.fom("Receivers").Offset = 1e9

chan.fom.State = 1
```

The following example can be run ONLY on a PNA with a second internal source, has revision A.07.10 or later, and has FOM (opt 080). It uses the internal 2nd source for the fixed LO frequency.

```
set app = CreateObject("Agilentpna835x.application")

set chan = app.ActiveChannel

chan.startFrequency = 1e9

chan.StopFrequency = 2e9

' set the receiver frequencies to be 2e9->3e9

chan.fom("Receivers").Offset = 1e9

chan.fom("Source2").Coupled = 0

chan.fom("Source2").StartFrequency = 1e9

chan.fom("Source2").StopFrequency = 1e9

' turn off port coupling

chan.coupleports = 0

' set LO to 10 dBm

chan.TestPortPower(3) = 10

'Turn ON port 3, our LO signal on our 2 source PNA

chan.SourcePortMode(3) = 1

chan.fom.State = 1
```

Last Modified:

8-Oct-2007 MX New topic

Limit Line Testing with COM

This Visual Basic program:

- Turns off existing Limit Lines
- Establishes Limit Lines with the following settings:
 - Frequency range - 4 GHz to 8 GHz
 - Maximum value - (10dB)
 - Minimum value - (-30dB)
- Turns on Lines, Testing, and Sound

If using [Global Pass/Fail](#) to report limit results, trigger the PNA after configuring and enabling Limit lines.

```
Public limts As LimitTest
Set limts = meas.LimitTest
'All Off
For i = 1 To 20
    limts(i).Type = naLimitSegmentType_OFF
Next i

'Set up Limit Lines
limts(1).Type = naLimitSegmentType_Maximum
limts(1).BeginResponse = 10
limts(1).EndResponse = 10
limts(1).BeginStimulus = 4000000000#
limts(1).EndStimulus = 8000000000#
limts(2).Type = naLimitSegmentType_Minimum
limts(2).BeginResponse = -30
limts(2).EndResponse = -30
limts(2).BeginStimulus = 4000000000#
limts(2).EndStimulus = 8000000000#

'Turn on Lines, Testing, and Sound
limts.LineDisplay = 1
limts.State = 1
limts.SoundOnFail = 1
```

Modify Display Colors

This VBScript example modifies display colors, modifies trace1 colors, then saves and recalls the theme.

These programs can be run as a macro in the PNA. To do this, copy the code into a text editor file such as Notepad and save on the PNA hard drive as Colors.vbs. [Learn how to setup and run the macro.](#)

```
function RGB(R, G, B)

    RGB = R + G*(2^8) + B*(2^16)

end Function

shell.AppActivate "PNA Series Network Analyzer"

Set app = CreateObject("AgilentPNA835X.Application")

app.preset

Set colors = app.Preferences.DisplayColors

' Uncomment the following line to modify Print colors
' Set colors = app.Preferences.PrintColors

colors.ResetTheme( )

colors.background      = RGB(64,0,64)      ' purple
displaycolors.grid    = RGB(0,255,128)    ' greenish
colors.activeLabels   = RGB(0,0,255)      ' blue
colors.inactiveLabels = RGB(255,0,0)      ' red
colors.failedTraces   = RGB(255,128,64)    ' orange

dim Trace1

Set Trace1 = colors.Trace(1)

Trace1.DataAndLimits = RGB(1,251,1)      ' green
Trace1.Memory         = RGB(251,1,1)      ' red
Trace1.Markers        = RGB(251,251,251)  ' white
```

```
Tracel.MemoryMarkers = RGB(1,251,251)    ' green + blue
```

```
colors.StoreTheme("c:/Program Files/Agilent/Network  
Analyzer/Colors/Themel.colors")
```

```
colors.LoadTheme("c:/Program Files/Agilent/Network  
Analyzer/Colors/Themel.colors")
```

E5091Testset Control

The following VB Script example exercises the COM commands used to control the E5091A testset.

For a description of each command, see [E5091Testsets collection](#).

```
Sub Main()  
Set pna = CreateObject("AgilentPNA835x.Application")  
Dim testsets As E5091Testsets  
Set testsets = pna.E5091Testsets  
Dim tset1 As E5091Testset  
Set tset1 = testsets(1)  
tset1.OutputPort(1, 3) = naE5091PortR2  
tset1.ControlLines(1) = 5  
tset1.ShowProperties = True  
tset1.Enabled = True  
MsgBox tset1.ID  
MsgBox tset1.Enabled  
MsgBox tset1.ShowProperties  
' NumberOfPorts property returns 0 when testset not connected  
MsgBox tset1.NumberOfPorts  
MsgBox tset1.OutputPort(1, 3)  
MsgBox tset1.ControlLines(1)  
  
Dim tset2 As E5091Testset  
Set tset2 = testsets(2)  
tset2.Enabled = True  
tset2.ShowProperties = True  
MsgBox tset2.Enabled  
MsgBox tset2.ShowProperties  
End Sub
```

Errors and the SCPIStringParser Object

This C++ program uses the [SCPIStringParser.Parse](#) command to detect the failed HRESULT and interrogate the errorInfo object for more details.

```
// scpierrors.cpp : Defines the entry point for the console
application.

//

#include <iostream>

#include "afx.h"

#include "atlbase.h"

#import "C:/program files/common files/agilent/pna/835x.tlb"
raw_interfaces_only, no_namespace, named_guids

using namespace std;

HRESULT SendScpiCommand( IScpiStringParser* parser, CComBSTR& cmd,
CComBSTR& response)
{
    CComBSTR bstr;

    HRESULT hr = parser->Parse(CComBSTR(cmd), &response);

    if (FAILED(hr))
    {
        // see if this interface supports ErrInfo
        CComPtr<ISupportErrorInfo> spSupportsErrInfo;

        if (SUCCEEDED(parser->QueryInterface(&spSupportsErrInfo)))
        {
            // it does, so let's get the errorinfo object
            CComPtr<IErrorInfo> spErrorInfo;
```

```

        if (SUCCEEDED(GetErrorInfo(0, &spErrorInfo))
        {
            CComBSTR errStr;

            spErrorInfo->GetDescription(&errStr);

            std::cout << "ERROR: " << CString(errStr) <<
            std::endl;

        }
    }
}

return hr;
}

int main()
{
    CoInitialize(NULL);
    {
        CComBSTR response;
        CComPtr<IApplication> spPNA;
        CComPtr<IScpiStringParser> spSCPI;

        if
        (SUCCEEDED(spPNA.CoCreateInstance(CLSID_Application)))
        {
            spPNA->get_ScpiStringParser(&spSCPI);

            SendScpiCommand(spSCPI, CComBSTR("SYSTEM:PRESET"),
            response);

            SendScpiCommand(spSCPI, CComBSTR("CALC:PAR:CAT?"),
            response);

            std::cout << CString(response) << std::endl;

            SendScpiCommand(spSCPI,

```

```
        CComBSTR( "THIS:IS:A:SYNTAX:ERROR" ), response);  
    }  
    }  
    CoUninitialize();  
    return 0;  
}
```

External Testset Control

The following VB Script example exercises the COM commands used to control the Z5623AK64 testset.

For a description of each command, see [TestsetControl Object](#)

See Other COM Example Programs

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save on the PNA hard drive as Testset.vbs. [Learn how to setup and run the macro.](#)

```
' Demonstrate some COM commands for external testsets.

Dim pna

Set pna = CreateObject("AgilentPNA835x.Application")

Sub DemoTestset(na)

Dim testsets, tset1

Dim portNum

Dim chNum, address

Set testsets = na.ExternalTestsets

chNum = 1

' Load a configuration file.

' NOTE: the K64 testset is only compatible with 4-port analyzers.

address = 0

testsets.Add "Z5623AK64", address

' Get the testset object

' in the testsets collection.

Set tset1 = testsets(1)
```

```
' Show the selections available for each port.

For portNum = 1 To 4
MsgBox("Port " & CStr(portNum) & " catalog: " & tset1.PortCatalog(portNum))
Next

' Set port mappings on channel 1.
tset1.OutputPorts(chNum) = "5 ext R,2 int R,3 int R,6 int R"

' Set control lines.
tset1.ControlLines(chNum) = 85

' Set label.
tset1.Label(chNum) = "Some label"

' Enable external testset control. This automatically enables status bar display
as well.
tset1.Enabled = True

End Sub

' The testset used in this demo is only usable on 4-port analyzers
If (pna.NumberOfPorts <> 4) Then
MsgBox("This program only runs on 4-port analyzers.")
Else
DemoTestset(pna)
End If
```

RF PathConfiguration Example

Note: These commands are accessible only for PNA-X models.

These Visual Basic and [C# examples](#) exercise various commands on the:

- [PathConfigurationManager Object](#)
- [PathConfiguration Object](#)
- [PathElement Object](#)

See Also

[IFPathConfiguration Setup](#) example

VB Example

```
' Create / Get the PNA application
Dim app
Set app = CreateObject("AgilentPNA835x.Application")
' Preset the instrument
app.Preset
' Get a channel interface on which to operate
Dim chan
Set chan = app.ActiveChannel
' Modify the Default configuration, and save it as "My Config"
chan.PathConfiguration = "Default"
' Set the "Combiner" element to value "Reversed"
chan.PathConfiguration.Element("Combiner").Value = "Reversed"
' Set the "Src1" element to value "High Power"
chan.PathConfiguration.Element("Src1").Value = "High Power"
' Change the description text
chan.PathConfiguration.DescriptionText = "Connect J8 to J9."
```

```

' Store the modified configuration
chan.PathConfiguration.Store ("My Config")

' Set the instrument's path config back to the default (req. 8)
chan.PathConfiguration = "Default"

' Load a previously saved configuration onto channel 1
app.PathConfigurationManager.Load 1, "My Config"

```

C# Example

```

Type pnaType =
Type.GetTypeFromProgID("AgilentPNA835x.Application", "PNA-NAME-
HERE");

    AgilentPNA835x.Application pna =
(AgilentPNA835x.Application)Activator.CreateInstance(pnaType);

    AgilentPNA835x.Channel chan =
(AgilentPNA835x.Channel)pna.ActiveChannel;

    // Preset the Instrument
    pna.Preset();

    // Modify the Default configuration, and save it as "My Config"
    chan.set_PathConfiguration("Default");

    // Set the "Combiner" element to value "Reversed"
    chan.get_PathConfiguration().get_Element("Combiner").Value =
"Reversed";

    // Change the description text
    chan.get_PathConfiguration().DescriptionText = "Connect J8 to
J9.";

    // Store the modified configuration
    chan.get_PathConfiguration().Store("My Config");

    // Set the instrument's path config back to the default (req.
8)
    chan.set_PathConfiguration("Default");

```

```
// Load a previously saved configuration onto channel 2
```

```
pna.PathConfigurationManager.LoadConfiguration(1, "My Config");
```

Last Modified:

6-Oct-2009 Fixed last two lines (ch1) and Add C#

Create a Narrowband Pulsed Measurement using the PNA-X or N522x

The following COM example demonstrates how to create a narrowband pulsed measurement using the Pulsed Application DLL on the [PNA-X](#).

See the example program for [wideband pulsed measurements](#) on the PNA-X.

It first gets valid configuration settings and then uses those settings to configure the PNA and internal pulsed generators.

To run this program, you need:

- PNA-X or N522x
- [Pulsed Application](#) (Option H08)

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Pulse.vbs. [Learn how to setup and run the macro.](#)

See Also

- Learn how to [install and register the pulsed .dll](#) on your PC
- See the [ConfigEnhancedNB2](#) method for sending and returning parameters to the .dll.
- See the [ConfigEnhancedNBIFAtten](#) method for setting the receiver IF gain.
- See the [COM IF Configuration](#) commands used in the program.
- See the equivalent [SCPI IF Configuration](#) commands.

```
'Interfaces

Dim OApp As AgilentPNA835x.application

Dim OIntPG As AgilentPNA835x.PulseGenerator

Dim OPathConf As AgilentPNA835x.PathConfiguration

Dim OFilter As AgilentPNA835x.SignalProcessingModuleFour

Dim OIF As AgilentPNA835x.IFConfiguration

'Pulsed parameters

Dim DPRF As Double

Dim DBW As Double
```

```
Dim DPhysicalIF As Double

Dim DNCO As Double

Dim DCF As Double

Dim DGD As Double

Dim DGW As Double

Dim DSWGD As Double

Dim DSWGW As Double

Dim DSWGR As Long

Dim LStage1TapArray() As Long

Dim LStage2TapArray() As Long

Dim LStage3TapArray() As Long

Dim BFixedPRF As Boolean

Dim IIFatten As Integer

'pulsed DLL interface

Dim OPulsed As New AgilentPNAPulsed.application

'Pulsed settings

DPRF = 5000 'Hz

DBW = 500 'Hz

BFixedPRF = True

DNCO = 0#

DCF = 0#

DGD = 0#

DGW = 0.000001

DSWGR = 0#

'Send desired pulsed parameters to the pulsed configuration DLL. The DLL will return
a new set of pulse parameters to send to the PNA.

OPulsed.ConfigEnhancedNB2 DPRF, DBW, DPhysicalIF, DNCO, DCF, LStage1TapArray,
LStage2TapArray, LStage3TapArray, BFixedPRF, DGD, DGW, DSWGD, DSWGW, DSWGR
```

```
'  
  
'Send configuration to PNA  
  
'  
  
'Connect to the PNA application  
Set OApp = CreateObject("AgilentPNA835x.Application")  
  
'Create instance of pulse generators on active channel  
Set OIntPG = OApp.ActiveChannel.PulseGenerator  
  
'Create instance of path configuration on active channel  
Set OPathConf = OApp.ActiveChannel.PathConfiguration  
  
'Create instance of digital filter on active channel  
Set OIF = OApp.ActiveChannel.IFConfiguration  
  
'Create instance of Hana digital filter on active channel  
Set OFilter = OApp.ActiveChannel.SignalProcessingModuleFour  
  
'Set up master pulse period for internal pulse generators  
OIntPG.Period = 1 / DPRF  
  
'Set up internal pulse generator output #1 to drive internal source modulation  
OIntPG.Width(1) = 0.0001 '100us  
OIntPG.Delay(1) = 0.00001 '10us  
OIntPG.State(1) = True  
  
OPathConf.Element("PulseModDrive").Value = "Pulse1"  
  
'Set up internal pulse generator output #2 to drive internal receiver gates for a 2  
port PNA-X  
  
OIntPG.Width(2) = 0.000001 '1us  
OIntPG.Delay(2) = 0.00005 '50us  
OIntPG.State(2) = True  
  
OPathConf.Element("IFGateA").Value = "Pulse2"  
OPathConf.Element("IFGateB").Value = "Pulse2"  
OPathConf.Element("IFGateR1").Value = "Pulse2"
```

```
OPathConf.Element("IFGateR2").Value = "Pulse2"

'
'Configure PNA in pulsed mode operation
'
'Turn off ALC and turn on modulator control
OApp.ActiveChannel.ALCLevelingMode(1) = naALCOpenLoop 'Source 1 output #1 ALC off
OPathConf.Element("Src1Out1PulseModEnable").Value = "Enable" 'Enable Source 1 pulse
modulator

'Set path and enable IF gates
OApp.ActiveChannel.IFBandwidth = DBW
OPathConf.Element("IFSigPathAll").Value = "NBF"

'Set filter stages based on pulse parameters
OIF.IFFrequency = DPhysicalIF
OIF.IFFrequencyMode = naMANUAL
OFilter.Stage1Frequency = DNCO
OFilter.Stage1Coefficients = LStage1TapArray
OFilter.Stage2Coefficients = LStage2TapArray
OFilter.Stage3FilterType = "RECT"
OFilter.Stage3Parameter("C") = LStage3TapArray(0)
OFilter.FilterMode = naMANUAL

'Set receivers to auto gain setting
OPulsed.ConfigEnhancedNBIFatten DPRF, DGW, IIFatten '1us pulse width
OPathConf.Element("NBFATNA").Value = IIFatten
OPathConf.Element("NBFATNB").Value = IIFatten
OPathConf.Element("NBFATNR1").Value = IIFatten
OPathConf.Element("NBFATNR2").Value = IIFatten
MsgBox "Done"
```

Last Modified:

27-Jul-2012	Some minor edits
11-Jun-2007	Rev 2- some edits
16-Feb-2007	MX New topic

Setup Basic Measurements

This VBScript program sets up four basic s-parameter measurements in four windows, all in a single channel. Handles are created to the measurement, channel, and window objects so that subsequent settings can be made for each.

Note: This is only an example. This is not necessarily the most efficient way to make basic S-parameter measurements.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Basic.vbs. [Learn how to setup and run the macro.](#)

[See PNA Object Model](#)

See [CreateSParameterEx](#)

See Other COM Example Programs

```
Set pna = CreateObject("AgilentPNA835x.Application")

pna.Preset

' Get a handle to the preset channel 1, S11 meas, and window(1)
set meas1=pna.ActiveMeasurement
set chan1=pna.ActiveChannel
set win1=pna.ActiveNAWindow

' Creates a new S21 measurement in New window(2)
pna.CreateSParameterEx 1,2,1,2,2
set meas2=pna.ActiveMeasurement
set win2=pna.ActiveNAWindow

' Creates a new S12 measurement in New window(3)
pna.CreateSParameterEx 1,1,2,1,3
set meas3=pna.ActiveMeasurement
set win3=pna.ActiveNAWindow

' Creates a new S22 measurement in New window(4)
```

```
pna.CreateSParameterEx 1,2,2,2,4
set meas4=pna.ActiveMeasurement
set win4=pna.ActiveNAWindow
'Make settings
'set Stop Frequency for channel
chan1.StopFrequency=1e9
'set Display formats
meas1.format=1 'Lin Mag
meas2.format=2 'Log Mag
meas3.format=3 'Phase
meas4.format=4 'Smith
'Show title in all windows
win1.title="Win #1"
win2.title="Win #2"
win3.title="Win #3"
win4.title="Win #4"
```

Last Modified:

19-Apr-2010 Modified example for same channel

Setup Compression Marker

This example program does the following:

- Creates a compression marker
- Queries the Power Out and Power In values

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as CompMkr.vbs. [Learn how to setup and run the macro.](#)

[See the FIFO object.](#)

See Other COM Example Programs

```
Set app = CreateObject("AgilentPNA835X.Application")
set meas = app.activemeasurement
'get the COM marker object
'and create marker1
set mark = meas.marker(1)
'set the compression level
mark.compressionlevel = 1.5
'make it a compression marker
'and find the compression point
mark.searchcompressionpoint
'return power out and power in
'power in
dim answer
answer = mark.compressionpin
wscript.echo("pin: " & answer)
'power out
answer = mark.compressionpout
```

```
wscript.echo("pout: " & answer)
```

Last Modified:

12-Feb-2009 MX New topic

Set Up an Embedded LO Measurement

This VBScript example creates a Noise Figure Converter measurement for a converter with an Embedded LO.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Noise.vbs. [Learn how to setup and run the macro.](#)

```
option explicit

dim app

dim chan

dim host

set app = CreateObject("Agilentpna835x.application")

app.reset

' create NFX traces

app.CreateCustomMeasurementEx 1, "Noise Figure Converters", "NF", 1

app.CreateCustomMeasurementEx 1, "Noise Figure Converters", "SC21", 1

'set channel and application objects

set chan = app.ActiveChannel

dim nfx

set nfx = chan.CustomChannelConfiguration

dim converter

set converter = chan.GetConverter()

dim calMgr

set calMgr = app.GetCalManager

dim nfxCal

set nfxCal = CalMgr.CreateCustomCalEx(1)

dim nfxCalExt

set nfxCalExt = nfxCal.CustomCalConfiguration

dim ELO
```

```
set ELO = converter.ConverterEmbeddedLO
' Set embedded LO properties
ELO.NormalizePoint = 101
ELO.TuningMode = 0 ' Broadband and precise
ELO.TuningIFBW = 3.0e4
ELO.MaxPreciseTuningIterations = 5
ELO.PreciseTuningTolerance = 1
ELO.TuningSweepInterval = 1
ELO.IsOn = true
'The following single sweep performs the same
'function as "Find Now" on the ELO dialog
chan.Single 1
```

Last Modified:

16-Aug-2012	Added Find now comment
27-Oct-2009	MX New topic

Setup FastCW and FIFO

This example program does the following:

- Setup an A/R and B/R measurement
- Turn ON point averaging
- Set external edge triggering (commented out)
- Set FIFO and Fast CW
- Write data into FIFO data buffer
- Read FIFO data buffer

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as FIFO.vbs. [Learn how to setup and run the macro.](#)

[See the FIFO object.](#)

See Other COM Example Programs

```
dim app
set app = createobject("Agilentpna835x.application")
' Setup and measure A/R and B/R
app.Reset
app.CreateMeasurement 1,"A/R1",0
app.CreateMeasurement 1,"B/R1",0
' Set IFBW to 600khz (400thousand pts/second)
app.activeChannel.IFBandwidth = 600e3
' Point Averaging Count = 10
app.activeChannel.AverageMode = 0 ' point
app.activeChannel.averagingFactor = 10
app.ActiveChannel.averaging = 1 ' turn on
' Edge triggering - positive edge
```

```
'app.TriggerSetup.ExternalTriggerConnectionBehavior(1) = 2 ' BNC1 = trigger
positive edge

'app.TriggerSetup.Source = 2 ' external

'app.ActiveChannel.TriggerMode = 0 'point

' Setup FIFO and Fast CW count

app.ActiveChannel.Hold 1 ' hold - synchronous

app.FIFO.State = 1' turn on FIFO

app.FIFO.Clear

app.activechannel.sweeptype = 3 ' CW sweep

app.activechannel.fastcwpointcount = 1000000' set the point count to 1million

app.activechannel.single 1 ' synchronous single

' the single will wait until the end of sweep.

'You do not have to wait until end of sweep to start emptying FIFO.

points = app.fifo.datacount

msgbox points

' points == 2000000 ' points = 2million. Took 5 seconds to acquire

For I = 0 to 1 ' 2 iterations (2 parameters * 2 sets of 1 million)

Dim data

Data = app.fifo.data(1000000)

Next

msgbox data(0)
```

Last Modified:

10-Oct-2008 MX New topic

Setup Noise Figure Port Mapping

This program demonstrates how to change source and receive ports when measuring noise figure. It assumes that option 029 ("Fully Corrected Noise Figure") is installed.

If only option 028 ("Noise figure measurements using standard receivers") is installed, switching ports is simpler, since only one noise receiver selection is available.

This program can be run as a macro in the PNA. To do this, copy the code into a text editor file such as Notepad and save on the PNA hard drive as NF.vbs. [Learn how to setup and run the macro.](#)

See Also

[Noise Figure Object](#)

[Create and Cal a NoiseFigure Measurement](#)

See Other COM Example Programs

```
option explicit
' Noise receiver enumerations
dim naStandardReceiver, naNoiseReceiver
' standard PNA receiver
naStandardReceiver = 0
' dedicated noise receiver (option 029 only)
naNoiseReceiver = 1
dim pna, windowNum, channelNum
set pna = CreateObject("Agilentpna835x.application")
windowNum = 1
channelNum = 1
pna.Reset
' Create Noise Figure measurement
dim noise, noiseChan, noiseConfig
```

```
set noise = pna.CreateCustomMeasurementEx(channelNum, "Noise  
Figure Cold Source", "NF", windowNum)  
  
set noiseChan = pna.ActiveChannel  
  
' provides access to noise-specific channel properties  
  
set noiseConfig = noiseChan.CustomChannelConfiguration  
  
'To change from the default input/output port settings of  
  
' source port = PNA1, receive port = PNA2,  
  
' you must first change the noise receiver,  
  
' then select the desired ports.  
  
dim srcPort, rcvPort  
  
' set port mapping to source port = PNA3, receive port = PNA4  
  
srcPort = 3  
  
rcvPort = 4  
  
' use PNA receiver for noise measurements  
  
noiseConfig.NoiseReceiver = naStandardReceiver  
  
noiseConfig.SetPortMap srcPort, rcvPort  
  
' To revert back to using the noise receiver, the source  
  
' and receive ports must be set to their default values  
  
' BEFORE switching to the noise receiver.  
  
' Otherwise, a COM exception will be thrown.  
  
' restore defaults: source=PNA1, receiver=PNA2  
  
noiseConfig.SetPortMap 1,2  
  
' use dedicated noise receiver for noise measurements  
  
noiseConfig.NoiseReceiver = naNoiseReceiver
```

Setup Phase Control

The following VB Script example exercises the COM commands used to setup and display Phase Sweep measurements.

See Also

[About Phase Control](#)

[PhaseControl Object](#)

See Other COM Example Programs

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save on the PNA hard drive as RxLevel.vbs. [Learn how to setup and run the macro.](#)

```
'Assume port 1 is connected to port 3

Set pna = CreateObject("AgilentPNA835x.Application")

pna.Preset

Set chan = pna.ActiveChannel
chanNum = chan.ChannelNumber

'Create 3 traces: S33, R3/C(amp),R3/C(phase)

pna.CreateMeasurement 1,"S33",3

Set meas1 = pna.ActiveMeasurement

meas1.Format = 4      'Smithchart format

pna.CreateMeasurement 1,"R3/C",3 'Log format

Set meas2 = pna.ActiveMeasurement

meas2.Format = 1      'Phase format

pna.CreateMeasurement 1,"R3/C",3

Set meas = pna.ActiveMeasurement

meas.Format = 2      'Phase format

'turn on 3 and 1
```

```
chan.SourcePortMode(1) = 1
chan.SourcePortMode(3) = 1
chan.SweepType = 5 'Phase sweep
Set phase = chan.PhaseControl
'set port3's control parameter to R3/C
phase.PhaseParameter(3) = "R3/C"
'notice the reference port should not included in the parameter
phase.PhaseReferencePort(3) = 1
'Set port3 to PAR mode
phase.PhaseControlMode(3) = 1 'PhaseControlParamter mode
phase.FixedRatioedPower(3) = 3
phase.StartPhase(3) = 0
phase.StopPhase(3) = 180
```

Last Modified:

28-Jan-2011 MX New topic

Setup PNOP and PSAT Marker Search

This example program does the following:

- Sets up measurement for either PNOP or PSAT marker search
- Sets parameters for search
- Reads a parameter for each

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as SearchMkr.vbs. [Learn how to setup and run the macro.](#)

See Other COM Example Programs

```
Set app = CreateObject("AgilentPNA835X.Application")

app.Preset

set meas = app.activemeasurement

'View Power Out vs Power In
meas.ChangeParameter "B",1

'perform power sweep

set chan = app.ActiveChannel

chan.SweepType = 2

chan.StartPower = -5

chan.StopPower = 0

'-----

'Choose marker search

resp=Msgbox ("PNOP (yes) or PSAT (no)" , 4, "PNA Marker Search Demo")

if resp=6 then

    PNOP1()

Else

    PSAT1()
```

```

End If

'-----

'PSAT marker search

Sub PSAT1()

set psat = meas.PSaturation

psat.PMaxBackOff = .3

psat.SearchPowerSaturation

'Read PSAT Parameter

dim answer

answer=psat.GainSaturation

wscript.echo("Gain Sat: "& answer)

End Sub

'-----

'PNOP marker search

Sub PNOP1()

set pnop = meas.PNOP

pnop.BackOff = 2

pnop.PinOffset = 1

pnop.SearchPowerNormalOperatingPoint

'Read PNOP Parameter

dim answer

answer=pnop.Gain

wscript.echo("PNOP Gain: "& answer)

End Sub

```

Last Modified:

19-Feb-2010 MX New topic

Setup Receiver Leveling

The following VB Script example exercises the COM commands used to setup Receiver Leveling.

See Also

[About Receiver Leveling](#)

[RxLevelingConfiguration Object](#)

See Other COM Example Programs

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save on the PNA hard drive as RxLevel.vbs. [Learn how to setup and run the macro.](#)

```
' Demonstrate some COM commands for Receiver Leveling.

Dim pna

Set pna = CreateObject("AgilentPNA835x.Application")

Dim chan

Set chan = pna.ActiveChannel

Dim RxLevel

Set RxLevel = chan.GetRxLevelingConfiguration

Dim srcPort

srcPort = 1

pna.Preset

RxLevel.ReferenceReceiver(srcPort) = "R1"

RxLevel.Tolerance(srcPort)= 0.02

RxLevel.IterationNumber(srcPort)= 10

RxLevel.FastMode(srcPort)=True

RxLevel.LevelingIFBW(srcPort)= 100

RxLevel.PowerOffset(srcPort)= 0

RxLevel.PowerMax(srcPort)= 20
```

```
RxLevel.PowerMin(srcPort)= -50
```

```
RxLevel.SafeMode(srcPort)= True
```

```
RxLevel.State(srcPort)= True
```

Last Modified:

1328-Jan-2011 MX New topic

Show Custom Cal Windows during a Guided Calibration

This VBScript program shows how to send commands that allow you to view specific 'custom' windows, and sweep specific channels, during a UI (Cal Wizard) or remote calibration.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as CalWindow.vbs. [Learn how to setup and run the macro.](#)

These commands are used to show and sweep specific windows and channels:

- [AllowChannelToSweepDuringCalAcquisition Method](#)
- [DisplayNAWindowDuringCalAcquisition Method](#)
- [DisplayOnlyCalWindowDuringCalAcquisition Method](#)
- [SweepOnlyCalChannelDuringCalAcquisition Method](#)

The following command sweeps the Cal Windows before remote acquisition:

- [SetupMeasurementsForStep Method](#)

See Other COM Example Programs

```
Set pna = CreateObject("AgilentPNA835x.Application")
pna.Preset
' get a handle to the preset channel 1 so that we can later cal it
set meas=pna.ActiveMeasurement
' Creates a new S21 measurement in channel 2 and New window(2)
' this will be the channel and window to show during cal
pna.CreateSParameterEx 2,2,1,2,2
Set calMgr = pna.GetCalManager
Set guidedCal = calMgr.GuidedCalibration
' show window 2 during cal
calMgr.DisplayNAWindowDuringCalAcquisition 2,True
```

```

' sweep channel 2 during calibration of chan 1
calMgr.AllowChannelToSweepDuringCalAcquisition 1,2,True
' make Channel1 the active channel
' activating the measurement also activates the channel
meas.Activate
guidedCal.Initialize 1, True
' Do 2-port cal
' Select the connectors
guidedCal.ConnectorType(1) = "APC 3.5 female"
guidedCal.ConnectorType(2) = "APC 3.5 male"
' Select the Cal Kit for each port being calibrated.
guidedCal.CalKitType(1) = "85052D"
guidedCal.CalKitType(2) = "85052D"
' Initiate the calibration and query the number of steps
numSteps = guidedCal.GenerateSteps
' Measure the standards, compute and apply the cal
value = MsgBox("Number of steps is " + CStr(numSteps))
' Measure the standards
For i = 1 to NumSteps
step = "Step " + CStr(i) + " of " + CStr(numSteps)
strPrompt = guidedCal.GetStepDescription(i)
' Sweep the Cal window prior to standard acquisition
guidedCal.SetupMeasurementsForStep i
' prompt to connect standard
value = MsgBox(strPrompt, vbOKOnly, step)
' measure standard
guidedCal.AcquireStep i
Next

```

```
' Conclude the calibration
guidedCal.GenerateErrorTerms
MsgBox ("Cal is complete!")
' clear the Cal window and channel flags
calMgr.DisplayOnlyCalWindowDuringCalAcquisition
calMgr.SweepOnlyCalChannelDuringCalAcquisition
```

Last Modified:

1-Nov-2007 New topic

COM Events Example

This Visual Basic program shows how to monitor the end of sweep. The program will set sweep time to various amounts and BEEPs when sweep is completed. This method allows other processes to continue while waiting for end-of-sweep. This program stops after 10 loops.

Note: To avoid **Permission Denied** problems, this should be run on the PNA and not a PC. To run it from a PC both units must be "trusted" and on the same [domain/workgroup](#).

```
Option Explicit
Dim na As AgilentPNA835x.Application
Dim WithEvents naEvt As AgilentPNA835x.Application
Dim ch As AgilentPNA835x.Channel
Dim sweepComplete As Boolean

Private Sub Form_Load()

Dim N As Integer
Set na = CreateObject("AgilentPNA835x.application")
na.preset
Set ch = na.ActiveChannel
na.DisallowAllEvents          ' Turn off all events
Set naEvt = na                ' Enable event interrupts
Do
N = N + 1                    ' Loop counter
ch.sweepTime = 1 + (Rnd * 9) ' Set random sweep-time from 1-10 sec
sweepComplete = False
ch.Single False              ' Trigger sweep
naEvt.AllowEventCategory naEventCategory_CHANNEL, True ' Enable Channel event
Do
DoEvents                    ' Allows other processes to continue
Loop Until sweepComplete = True
naEvt.AllowEventCategory naEventCategory_CHANNEL, False ' Disable event until
ready for next one
Beep                        ' Do end-of-sweep processing here;

Loop Until N > 10
End

End Sub

Private Sub naEvt_OnChannelEvent(ByVal eventID As Variant, ByVal chNumber As
Variant)
' In this example we don't care about the channel info
If eventID = naEventID_CHANNEL_TRIGGER_COMPLETE Then sweepComplete = True
```

End Sub

Upload a Segment Table using C++

This example program shows the Variant method for uploading a segment sweep to the PNA using the [SetAllSegments](#) method

```
#include "stdafx.h"

#include <stdio.h>

#include "atlbase.h"

#include "objbase.h"

// import the PNA type library
//-----

#import "C:/Program Files/Common Files/Agilent/PNA/835x.tlb" no_namespace,
named_guids

int _tmain(int argc, _TCHAR* argv[])
{

    // interface pointers to retrieve COM interfaces

    HRESULT hr;

    IUnknown* pUnk = 0;

    IApplication* pNA = 0;

    IChannel* pChan = 0;

    IMeasurement* pMeas = 0;

    IMeasurement5* pMeas5 = 0;

    IArrayTransfer* pTrans = 0;

    ISegments* pSeg = 0;

    ISegments2* pSeg2 = 0;
```

```

//Variables for X and Y data read portion

SAFEARRAY* sArray;

_variant_t vXVals;

double HUGEPR* xVals;

float* pScalarData;

//Variables for Segment portion

double Fstart, Fstop, SegWidth;

long i[2];

int num_points = 11;

SAFEARRAY* pSA;

VARIANT vSeg;

VARIANT v;

int NUM_SEGS = 10;

int SEG_SIZE = 7;

//Create SafeArray to hold the segment data

SAFEARRAYBOUND aDim[2]; //This must be 2 the PNA expects to see a 2 dimensional
array

aDim[0].lLbound = 0;

aDim[0].cElements = SEG_SIZE; //This will be set to 7 unless port power is
uncoupled

aDim[1].lLbound = 0;

aDim[1].cElements = NUM_SEGS;

pSA= SafeArrayCreate(VT_VARIANT,2,aDim); //The cDim parameter must be set to 2 as
the PNA expects a 2D array

//Init Variant to set values in Safearray

```

```

VariantInit(&vSeg);

Fstart=10e6;
Fstop=3e9;
SegWidth=(Fstop-Fstart)/NUM_SEGS;

//Loop to write segment data
for(int j=0; j<NUM_SEGS; ++j)
{
    i[1]=j; //Set Segment #

    //Segment Definition
    i[0] = 0;
    vSeg.vt = VT_BOOL; //First parameter is Boolean
    vSeg.boolVal = VARIANT_TRUE; //Segment State
    SafeArrayPutElement(pSA, i, &vSeg);
    i[0] += 1;
    vSeg.vt = VT_I4; //Second parameter is an integer
    vSeg.intVal = num_points; //Number of Points
    SafeArrayPutElement(pSA, i, &vSeg);
    i[0] += 1;
    vSeg.vt = VT_R8; //Remaining parameters are of type double
    vSeg.dblVal = Fstart+j*SegWidth; //Start Frequency
    SafeArrayPutElement(pSA, i, &vSeg);
    i[0] += 1;
    vSeg.dblVal=vSeg.dblVal+SegWidth; //Stop Frequency
    SafeArrayPutElement(pSA, i, &vSeg);
}

```

```

    i[0] += 1;

    vSeg.dblVal = 1.0e3; //IF Bandwidth
    SafeArrayPutElement(pSA, i, &vSeg);
    i[0] += 1;

    vSeg.dblVal = 0.0; //Dwell time
    SafeArrayPutElement(pSA, i, &vSeg);
    i[0] += 1;

    vSeg.dblVal = -5.0; //Power
    SafeArrayPutElement(pSA, i, &vSeg);
}

//vSeg no longer needed, clean up
VariantClear(&vSeg);

//Declare Variant to use with Segment data
VariantInit(&v);
v.vt = VT_ARRAY|VT_VARIANT;
v.parray = pSA; //write safearray to variant

// Initialize the COM subsystem
CoInitialize(NULL);

CoInitializeSecurity(NULL,          //security descriptor
    -1,          // authn svc entries
    NULL,        // authn svcs
    NULL,        // reserved
    RPC_C_AUTHN_LEVEL_NONE,
    RPC_C_IMP_LEVEL_IMPERSONATE,
    NULL,        // authn info

```

```

0,          // capabilities
NULL);     // reserved

// Create an instance of the network analyzer
// Request the NA's IUnknown interface

hr = CoCreateInstance(CLSID_Application,0,CLSCTX_ALL,IID_IUnknown, (void**)
&pUnk);

if (!FAILED(hr))
{
    // QueryInterface for the INetworkAnalyzer interface of the NetworkAnalyzer
    object

    hr = pUnk->QueryInterface(IID_IApplication,(void**)&pNA);

    if (!FAILED(hr))
    {
        // Reset the analyzer to instrument preset

        pNA->Reset();

        // Create S11 measurement

        pNA->CreateSParameter(1,1,1,1);

        // Set pChan variable to point to the active channel

        pNA->get_ActiveChannel(&pChan);

        //Show Segment table

        pNA->NAWindows->Item(1)->ShowTable((NATableType)2);

```

```
//Get handle to ISegments Interface
pChan->get_Segments(&pSeg);

//Get handle to ISegments2 Interface
hr = pSeg->QueryInterface(IID_ISegments2, (void**)&pSeg2);

//Set Segment Sweep Options
pSeg2->IFBandwidthOption = VARIANT_TRUE;
pSeg2->SourcePowerOption = VARIANT_TRUE;

//Push segments to PNA
pSeg2->SetAllSegments(v);

//Set Sweep Type to Segment Sweep
pChan->SweepType = naSegmentSweep;

if (pChan)
{
    // Set pMeas variable to point to the active measurement
    pNA->get_ActiveMeasurement(&pMeas);

    if(pMeas)
    {
        // Setup the channel for a single trigger
        pChan->Hold(true);
        pNA->TriggerSignal = naTriggerManual;
        pChan->TriggerMode = naTriggerModeMeasurement;
    }
}
```

```

// Make the PNA application visible
pNA->put_Visible(true);

// Send a manual trigger to initiate a single sweep
pChan->Single(true);

// QueryInterface for the IArrayTransfer interface of the
NetworkAnalyzer object
hr = pMeas->QueryInterface(IID_IArrayTransfer, (void**)&pTrans);

// Get handle for IMeasurement5 interface
hr = pMeas->QueryInterface(IID_IMeasurement5, (void**)&pMeas5);

if (!FAILED(hr))
{
    int val = num_points*NUM_SEGS;

    // Store the data in the "result" variable
    pScalarData = new float[val];
    xVals = new double[val*2];

    //Get X axis values
    vXVals=pMeas5->GetXAxisValues();

    //Convert _variant_t array to a SAFEARRAY
    sArray = vXVals.parray;

```



```
{  
    printf("Programmed failed to connect to the PNA.");  
}  
}  
CoUninitialize();  
  
system("PAUSE");  
return 0;  
}
```

Using C#

The following are common C# examples:

Connecting to a specific PNA via DCOM:

```
public AgilentPNA835x.Application Connect(string hostname)
{
    AgilentPNA835x.Application pna = null;
    try
    {
        Type t = Type.GetTypeFromProgID("AgilentPNA835x.Application", hostname,
true);
        pna = (AgilentPNA835x.Application) Activator.CreateInstance(t);
    }
    catch (Exception e)
    {
        HandleExceptions(e);
    }
    return pna;
}
```

Using the GetData Interface

```

AgilentPNA835x.IMeasurement meas = app.ActiveMeasurement;

    object[] dataArrayAsObj;

    dataArrayAsObj =
(object[])meas.getData(AgilentPNA835x.NADataStore.naMeasResult,
AgilentPNA835x.NADataFormat.naDataFormat_LogMag);

    float[] dataArray = new float[dataArrayAsObj.Length];

    for (int j = 0; j < dataArrayAsObj.Length; j++)
    {
        dataArray[j] = (float)dataArrayAsObj[j];
    }

```

2-dimensional GetData

```

AgilentPNA835x.IMeasurement meas = app.ActiveMeasurement;

    app.ActiveChannel.Single(true);

    object[,] dataArrayAsObj;

    dataArrayAsObj =
(object[,])meas.getData(AgilentPNA835x.NADataStore.naRawData,
AgilentPNA835x.NADataFormat.naDataFormat_Smith);

    float[,] dataArray = new float[dataArrayAsObj.Length,2];

    for (int j = 0; j < dataArrayAsObj.Length; j++)
    {
        dataArray[j,0] = (float)dataArrayAsObj[j,0];
        dataArray[j,1] = (float)dataArrayAsObj[j,1];
    }

```

Other C# / .NET Topics

[Perform a Guided Cal with CSharp](#)

[Getting a handle to the Noise Figure Cal object.](#)

[Using .NET](#)

Last Modified:

15-Sep-2009 Modified GetData example

Configure for COM-DCOM Programming

Before developing or running a COM program, you should first establish communication between your PC and the analyzer. This process is referred to as gaining **Access** to the analyzer. You should then register the PNA type library on your PC.

DCOM (Distributed Component Object Model) refers to accessing the PNA from a remote PC.

COM refers to accessing the PNA application from the analyzer PC.

- [Access Concepts](#)
- [Access Procedures](#)
- [Register the PNA Type Library on Your PC](#)
- [Problems?](#)

Note: After performing a [Firmware Upgrade](#) you must copy the new type library to your development PC to get access to new COM commands. See [Register the analyzer on your PC](#).

Other Topics about COM Concepts

For detailed information on this subject, see <http://na.tm.agilent.com/pna/DCOMSecurity.html>

Note: ONLY 32-bit compiler option is supported (64-bit is NOT supported).

Access Concepts

PNAs are shipped from the factory such that **Everyone** has permission to launch and access the PNA application via COM/DCOM. The term **Everyone** refers to a different range of users depending on whether the PNA is a member of a **Domain** or **Workgroup** (it must be one or the other; not both). By default, the PNA is configured as members of a workgroup. Therefore, **Everyone** includes only those users who have been given logon accounts on the PNA.

Workgroup

A workgroup is established by the **PNA administrator** declaring the workgroup name and declaring the PNA as a member of the workgroup. A workgroup does not require a network administrator to create it or control membership.

Everyone includes only those users who have been given logon accounts on the PNA.

By default, the PNA is configured as members of a workgroup named WORKGROUP.

Note: To setup a logon account for a new user, see [Additional Users](#).

The easiest method of gaining DCOM access, is to make the user's account name and password on the PNA to EXACTLY match their PC logon account name and password.

Domain

A domain is typically a large organizational group of computers. Network administrators maintain the domain and control which machines have membership in it.

Everyone includes those people who have membership in the domain. In addition, those with logon accounts can also access the analyzer.

Summary

- A **Workgroup** requires no maintenance, but allows DCOM access to only those users with a log-on account for the PNA.
- A **Domain** requires an administrator, but all members of the domain and those with logons to the analyzer are allowed DCOM access to the PNA.

The following section "Access Procedures" provides a tighter level of security allowing only **selected** (not **Everyone**) domain and workgroup users DCOM **Access** and **Launch** capability of the PNA.

Access Procedures

Perform this procedure for the following reasons:

- To allow only selected users (not everyone) remote Access and remote Launch capability to the PNA. Launch capability is starting the PNA application if it is not already open.
- To verify that you have DCOM access to the analyzer.

Note: Before doing this procedure, you must first have a logon account on the PNA. See [PNA User Accounts](#)

The following procedure grants specific users DCOM access and launch capability of the PNA application:

To perform this procedure, you must first [minimize the PNA](#) application.

[How do I know which Operating System I have?](#)

Windows 7
On the PNA, click the Windows Start button
Click Run
In the Open: box, type dcomcnfg
Click OK
Open the following folder sequence: Component Services Computers My Computer DCOM Config Right click Agilent PNA Series
Click Properties
Click the Security tab
Under Access Permissions , click Customize , then click Edit
Select Everyone , then click Remove
Click Add
Type a group name or user account name
Click OK
Under Launch Permissions , click Customize , then click Edit
Select Everyone , then click Remove
Click Add
Type a group name or user account name
Click OK

Register the PNA Type Library on Your PC

The type library contains the PNA object model. On your PC, there is a Registry file that keeps track of where object models are located. Therefore, you must register the type library on the PC that will be used to develop code and run the program. It is much more efficient to have the type library registered at design time (BEFORE running your COM program).

Do the following two items before proceeding:

1. Connect your PC and the PNA to LAN.
2. Either map a drive to the analyzer or copy the type library files on a floppy disk or other media. See [Drive Mapping](#).

Note: To register the type library on your PC, you must be logged on as an administrator of your PC. Learn about [User Accounts](#).

This procedure will do the following:

- Register the Network Analyzer application on your PC.
 - Copy and register the proxystub (835xps.DLL) onto the PC.
 - Copy and register the PNA type library (835x.tlb) onto the PC.
 - Copy and register the FCA type library (fca.tlb) onto the PC.
1. Using Windows Explorer on your PC, find the Analyzer's C: drive. The drive will not be named "C:" on your PC, but a letter you assigned when mapping the drive.
 2. Navigate to **Program Files / Agilent / Network Analyzer / Automation**
 3. Double-click **pnaproxy.exe** and follow the prompts to Install PNA Proxy. If the installation offers a choice of Modify, Repair, or Remove, then select **Remove**. Then double-click on **pnaproxy.exe** again.
 4. When prompted, type the Computer name of the PNA ([Learn how to find this](#)).
 5. After the install program runs, the PNA and FCA type library should be registered on your PC.
 6. Your programming environment may require you to set a reference to the PNA type library now located on your PC. In Visual Basic, click **Project, References**. Then browse to **C:/Program Files/Common Files/Agilent/PNA** Select **835x.tlb**

Problems?

- These procedures will fail if there are any programs using the PNA type library (for example: Visual basic, VEE, Visual Studio, or any other application program that may communicate with the PNA).
- Perform the following procedure if the previous procedure did not return an error, but you cannot connect to the PNA.
- If you received an error, check that both the account name and password used on both the PNA and PC match EXACTLY.
- If you still get errors, see <http://na.tm.agilent.com/pna/DCOMSecurity.html>.

1. [Map a drive](#) from your remote PC to the PNA. Note the drive letter your PC assigns to the PNA. Substitute this drive letter for **PNA** in the following procedure.
2. On your PC, go to a DOS prompt c:>
3. Type **PNA:** (for example o:)
4. Type **cd program files/agilent/network analyzer/automation**
5. Type **copy 835xps.dll c:/program files/common files/agilent/pna**
6. Type **copy 835x.tlb c:/program files/common files/agilent/pna**

If you will NOT be using [FCA commands](#), skip steps 7, 8, and 9.

7. Type **cd..**
8. Type **cd extensions/fca**
9. Type **copy fca.tlb c:/program files/common files/agilent/pna**
10. If it is not already there, copy **regtlib.exe** from **PNA:/WINNT** to your C:/<windows>/system32 directory (<windows> is OS-dependent- it is either windows or WINNT)
11. Type **regtlib C:/program files/common files/agilent/pna/835x.tlb**
12. Type **regsvr32 C:/program files/common files/agilent/pna/835xps.dll**
13. Type **regtlib C:/program files/common files/agilent/pna/fca.tlb**

Perform the [Access Procedure](#) after doing these steps.

COM Fundamentals

The following terms are discussed in this topic:

- [Objects](#)
- [Interfaces](#)
- [Collections](#)
- [Methods](#)
- [Properties](#)
- [Events](#)
- [Visual Basic Syntax](#)

Note: The information contained in this topic is intended to help an experienced SCPI programmer transition to COM programming. This is NOT a comprehensive tutorial on COM programming.

Other Topics about COM Concepts

Visual Basic Syntax

The examples in PNA Help use Visual Basic as the programming environment for COM, which uses 'dot' notation.

To set a property, follow the object reference with:

- a period (.)
- property or method
- an equal sign (=)
- the new value

For example:

```
object.property = value 'This Green text following an apostrophe (') is a comment.
```

To read a property, a variable to contain the returned value is followed with:

- an equal sign (=)
- an object, or reference to an object
- a period (.)
- property

For example:

```
variable = object.property
```

To execute a method, an object, or reference to an object is followed with:

- a period (.)
- the method
- a blank space
- any required parameters

For example:

```
object.method parameters
```

Some methods return values, such as methods that return data. To return data from a method, a variable to contain the returned data is followed with:

- an equal sign (=)
- an object, or reference to an object
- a period (.)
- the method
- any required parameters enclosed in parenthesis

```
variable = object.method (parameters)
```

Objects

The objects of the Network Analyzer (Application) are arranged in a hierarchical order. The [PNA object model](#) lists the objects and their relationship to one another.

In SCPI programming, you must first select a measurement before making settings. With COM, you first get a handle to the object (or collection) and refer to that object in order to change or read settings (properties).

For more information on working with objects, see [Getting a Handle to an Object](#).

Interfaces

A COM Interface is the connection to an object. When you get a handle to an object, you are actually using an interface to an object. This is important if you are developing PNA code that will run on multiple code versions. For more information, see [PNA Interfaces](#).

Collections

A collection is an object that contains several other objects of the same type. For example, the **Channels** collection contains all of the channel objects.

Note: In the following examples, the collections are referred to as a variable. Before using a collection object, you must first get an instance of that object. For more information, see [Getting a Handle to an Object](#)

Generally, items in a collection can be identified by **number** or by **name**. The order for objects in a collection cannot be assumed. They are always unordered and begin with 1. For example, in the following procedure, chans(1) is used to set averaging on the **first** channel in the Channels collection (not necessarily channel 1).

```
Sub SetAveraging()  
    chans(1).AveragingFactor = 10  
End Sub
```

The following procedure uses the measurement string name to set the display format for a measurement in the measurements collection.

```
meass("CH1_S11_1").Format = 1
```

You can also manipulate an entire collection of objects if the objects share common methods. For example, the following procedure sets the dwell time on all of the segments in the collection.

```
Sub setDwell()  
For Each seg In segs  
segs.DwellTime = 0.03  
Next  
End Sub
```

Methods

A method is an action that is performed on an object. For example, **CreateSParameter** is a method on

the [Application](#) object. The following procedure uses that method to create a new S21 measurement in channel 1 in a new window.

```
Sub CreatMeas  
app.CreateSParameter 1,2,1,1  
End Sub
```

Properties

A property is an attribute of an object that defines one of the object's characteristics, such as size, color, or screen location. A property can also change an aspect of the object's behavior, such as whether the object is visible. In either case, to change the characteristics of an object, you change the values of its properties.

For example, the following statement sets the IF Bandwidth of a channel to 1 KHz.

```
Chan.IFBandwidth = 1e3
```

You can also read the current value of a property. The following statement reads the current IF Bandwidth of a channel into the variable **Ifbw**.

```
Ifbw = Chan.IFBandwidth
```

Some properties cannot be set and some cannot be read. The Help topic for each property indicates if you can:

- Set and read the property (Write/Read)
- Only read the property (Read-only)
- Only set the property (Write-only)

Events

A COM event is an action recognized by an object, such as clicking the mouse or pressing a key. Using events, your program can respond to a user action, program code, or triggered by the analyzer.

The SCPI equivalent of an event is a Service Request (SRQ).

For example:

```
OnChannelEvent
```

For more information, see [Working with the Analyzer's Events](#).

Getting a Handle to an Object

The following are discussed in this topic:

- [What Is a Handle](#)
- [Declaring an Object Variable](#)
- [Assigning an Object Variable](#)
- [Navigating the Object Hierarchy](#)
- [Getting a Handle to a Collection](#)

Other Topics about COM Concepts

What Is a Handle

In SCPI programming, you must first select a measurement before changing or reading settings. With COM, you first get a handle to the object (or collection) and refer to that object to change or read its settings. The following analogy illustrates this:

A CAR could be called an object. More precisely, CAR is a class of objects. For example, one of the properties of the CAR class is "**Color**". You can read (by looking) or set (by painting) the color property of A car object. In other words, you can only read or set properties of a specific car object; not the entire car class. Therefore, to read or set a property, you need to get "a handle", or an instance of the object.

This process is also called "accessing an object", "getting an instance of an object", "returning an object". or "referring to an object". You can have handles to many instances of an object at the same time.

Accessing PNA Objects

The PNA Application object is the highest object in the PNA object model hierarchy. Because of that, it is the only object that must be 'created' before it, or any other objects, can be accessed and used. During the creation process, the application object assigned to a variable name, or handle. Throughout your program, that object is used by referring to that variable. All PNA objects can be assigned to a variable, and subsequently referred to, in this same manner.

The following example shows how to create the PNA Application object, as well as illustrate the general steps of get a handle to an object.

There are two steps in the process of getting a handle to analyzer objects:

1. Declaring a Variable
2. Assigning an Object to the Variable

1. Declaring a Variable

Note: The examples in these topics use the Visual Basic Programming Language. See the short section regarding [Visual Basic syntax](#).

Use the Dim statement or one of the other declaration statements (Public, Private, or Static) to declare a variable. The type of variable that refers to an object must be a Variant, an Object, or a specific type of object. Some programming languages, such as VBScript and Agilent VEE, do not allow you to specify variable types.

The following examples ALL declare the variable **pna**. Each subsequent statement is more specific than the previous:

- `Dim pna 'Variant data type.`
- `Dim pna As Object 'Object data type.`
- `Dim pna As AgilentPNA835x.Application ' Specific Application type`
- `Dim pna As AgilentPNA835x.IApplication ' Interface type`

1. If you use a variable without declaring it first, the data type of the variable is Variant. If you don't care about using automatic type checking, and willing to run code less efficiently, this method is very safe and is useable on all programming environments.
2. If you know the specific object type, and your programming environment allows it, you can declare the variable as an object.
3. Declaring a specific object type provides automatic type checking (Intellisense), faster code, and improved readability.
4. Declaring the interface is the most specific way and is beneficial when developing code for multiple firmware revisions. [Learn more about Interfaces.](#)

2. Assigning an Object to a Variable

To assign an object instance to a variable, use the **Set** keyword before the object variable that was declared previously. In the following line of code, we SET the current AgilentPNA835x Application to "pna".

```
Set pna = AgilentPNA835x.Application
```

As mentioned earlier, the AgilentPNA835x object is unique because it is the highest level of object in the PNA object model hierarchy. Therefore, we must use the **CreateObject** keyword with the (*classname,server name*) parameters.

- The **classname** for the analyzer object is always "AgilentPNA835x.Application".
- To find your analyzer's **server name**, see [View or change full computer name](#)

The following statements create an instance of the Analyzer object.

```
Dim pna AS AgilentPNA835x.Application
Set pna = CreateObject("AgilentPNA835x.Application", "Analyzer46")
```

Note: These statements will start the PNA application if it is not already running on your instrument.

Navigating the Object Hierarchy

Once an instance of the PNA Application is "created", you access all of the PNA objects by navigating the object hierarchy. Navigating the object model hierarchy can be tricky. In addition, you also need to know how to refer to a specific instance of that object. For example, if you have three measurements present on the PNA, how do you refer to the channel 1 measurement? Each object on the PNA Object Model image is linked to an object page. At the top of each object page is a **Description** section and another called "**Accessing the ... Object**". These sections together explain how to navigate the PNA hierarchy to access a specific instance of that object.

From the previous discussion, you may think that you must always declare and assign variables to an object before setting or reading its properties. While this method is best for objects that you will continue to reuse, such as a measurement, it is not always necessary. You can also refer to an object directly.

The [TriggerSetup](#) object, which is a child of the Application object. Because we will only need to refer to this object once to set a couple of properties, and it is easy to access, we will refer to it directly. From the previous example, we already have a handle to the Application object in the variable **pna**. The following example uses [Visual basic 'dot' notation](#) to refer to the TriggerSetup object, and then the Scope property.

```
pna.TriggerSetup.Scope = naChannelTrigger
```

By referring to the TriggerSetup object directly, we must type the same path whenever we refer to properties on the TriggerSetup object. The following method assigns the pna.TriggerSetup object to a variable that can be reused.

```
Dim trig As Object
Set trig = pna.TriggerSetup
```

Once created, you can treat an object variable exactly the same as the object to which it refers. For

example:

```
trig.Scope = naChannelTrigger
trig.Source = naTriggerSourceInternal
```

Getting a Handle to a Collection

The analyzer has several collections of objects which provide a convenient way of setting or reading all of the objects in the collection with a single procedure. Also, there are objects (limit lines for example) that can only be accessed through the collection.

To get a handle to an item in a collection, you can refer to the object by item number or sometimes by name. However, you first have to get a handle to the collection. To assign the collection to a variable, use the same two step process (1. declare the variable, 2. assign the variable using 'Set').

```
Dim meass As Measurements
```

```
Dim meas As Measurement
```

You can then iterate through the entire collection of measurements to read or set properties

```
Sub setFormat()
For Each meas In meass
    meas.Format = naDataFormat_LinMag
Next
End Sub
```

Or you can read or set a property on an individual object in the collection:

```
meass(1).Format = naLinMag
```

Note: Each object and collection has its own unique way of dealing with item names, and numbers. Refer to the [Analyzer Object Model](#) for details.

Collections in the Analyzer

Collections are a gathering of similar objects. They are a convenience item used primarily to iterate through the like objects in order to change their settings. Collections generally provide the following generic methods and properties:

```
Item(n)
Count
Add(n)
Remove(n)
```

where **(n)** represents the number of the item in the collection. Some collections may have unique capabilities pertinent to the objects they collect.

Other Topics about COM Concepts

Collections are Dynamic

A collection does not exist until you ask for it. When you request a Channels object (see Getting a Handle to an Object / [Collection](#)), handles to each of the channel objects are gathered and placed in an array.

For example, if channels 2 and 4 are the only channels that exist, then the array will contain only 2 items. The command 'channels.Count' will return the number 2, and:

- Channels(1) will contain the channel 2 object.
- Channels(2) will contain the channel 4 object.

The ordering of objects within the collection should not be assumed. If you add a channel to the previous example, as in:

```
Pna.Channels.Add(3)
```

'channels.Count' will now return 3 and:

- Channels(1) will contain the channel 2 object.
- Channels(2) will contain the channel 3 object.
- Channels(3) will contain the channel 4 object.

Primarily, collections are useful for making this type of iteration possible:

```
Dim ch as Channel
For each ch in pna.Channels
```

```
Print ch.Number  
Print ch.StartFrequency  
Print ch.StopFrequency  
Next ch
```

As soon as this for-each block has been executed, the Channels object goes out of scope.

COM Data Types

The PNA uses several data types to communicate with the host computer. Before using a variable, it is best to declare the variable as the type of data it will store. It saves memory and is usually faster to access. The following are the most common data types:

- [Long Integer](#)
- [Single Precision \(Real\)](#)
- [Double Precision \(Real\)](#)
- [Boolean](#)
- [String](#)
- [Object](#)
- [Enumeration](#)
- [Variant](#)

Other Topics about COM Concepts

Long (long integer) variables are stored as signed 32-bit (4-byte) numbers ranging in value from -2,147,483,648 to 2,147,483,647.

Double (double-precision floating-point) variables are stored as IEEE 64-bit (8-byte) floating-point numbers ranging in value from -1.79769313486232E308 to -4.94065645841247E-324 for negative values and from 4.94065645841247E-324 to 1.79769313486232E308 for positive values.

Single (single-precision floating-point) variables are stored as IEEE 32-bit (4-byte) floating-point numbers, ranging in value from -3.402823E38 to -1.401298E-45 for negative values and from 1.401298E-45 to 3.402823E38 for positive values.

Boolean variables are stored as 16-bit (2-byte) numbers, but they can only be True or False. Use the keywords True and False to assign one of the two states to Boolean variables.

When other numeric types are converted to Boolean values, 0 becomes False and all other values become True. When Boolean values are converted to other data types, False becomes 0 and True becomes -1.

In PNA release 5.26, the following properties were changed to return True rather than 1 to conform with this definition. This change may affect the functionality of your COM program:

- [Bandwidth Tracking Property](#)
 - [ErrorCorrection Property](#)
 - [Interpolate Correction Property](#)
 - [LimitTestFailed Property](#)
-

String variables hold character information. A String variable can contain approximately 65,535 bytes (64K), is either fixed-length or variable-length, and contains one character per byte. Fixed-length strings are declared to be a specific length. Variable-length strings can be any length up to 64K, less a small amount of storage overhead.

Object variables are stored as 32-bit (4-byte) addresses that refer to objects within the analyzer or within some other application. A variable declared as Object is one that can subsequently be assigned (using the Set statement) to refer to any actual analyzer object.

Enumerations (Enum) are a set of named constant values. They allow the programmer to refer to a constant value by name instead of by number. For example:

```
Enum DaysOfWeek
  Sunday = 0
  Monday = 1
  Tuesday = 2
  Wednesday = 3
  Thursday = 4
  Friday = 5
  Saturday = 6
End Enum
```

Given this set of enumerations, the programmer can then pass a constant value as follows:

```
SetTheDay(Monday)
```

rather than

```
SetTheDay(1)
```

where the reader of the code has no idea what the value 1 refers to.

However, the analyzer RETURNS a long integer, not the text.

```
Day = DaysOfWeek(today) 'Day = 1
```

Variant - If you don't declare a data type ("typed" data) the variable is given the Variant data type. The

Variant data type is like a chameleon - it can represent many different data types in different situations.

The PNA provides and receives Variant data because there are programming languages that cannot send or receive "typed" data. Variant data transfers at a slower rate than "typed" data.

PNA Interfaces

A COM interface is the connection to an object. When you [get a handle to an object](#), you are actually using an interface to an object. This subtle distinction is relevant to the COM programmer for the following two reasons:

- [Interface Inheritance \(Coding for Multiple PNA Versions\)](#)
- [Custom Interfaces.](#)

Other Topics about COM Concepts

Interface Inheritance (Coding for Multiple PNA Versions)

The PNA continues to evolve and release [new firmware / software versions](#) that provide more functionality and features. New commands are added to existing objects, and with them new interfaces are added to support those commands. For example, new commands were added to the Measurement object in PNA release 3.0. These commands are accessible from the new IMeasurement2 interface. This can be important if you develop code using the type library in release 3.0, and run the code on a PNA with an older release, such as 2.0

When you use a command that was new with release 3.0, and you run that code on a PNA with release 2.0 firmware, errors will occur because that PNA does not recognize the new commands. However, even if you do NOT utilize new commands, errors can still occur. The following example shows how this occurs and how to avoid it.

The following Visual Basic statement dimensions the **meas** variable as an object.

```
Dim meas As Measurement
```

When the program compiles, Visual Basic figures out what interface to use to access that object. When dimensioning as an object, VB will use the default interface. As new interfaces are added to an object, they become the default interface. If this program was developed and compiled using the PNA 3.0 type library, the default Interface of the Measurement Object was IMeasurement2. However, if this program is run on an instrument with PNA 2.0 firmware, there was no IMeasurement2 Interface, and an E_NOINTERFACE error will occur.

Therefore, the more robust approach would be to specify the interface instead of the object when declaring a variable.

```
Dim meas As IMeasurement
```

This code will ONLY use the IMeasurement interface; not the default interface.

However, regardless of how you declare a variable, errors will always occur if you use new commands, and run the code on an older instrument.

Custom Interfaces

The PNA object model contains three "custom" interfaces use "typed" variables, which is more efficient than using variant type variables. However, these interfaces are only usable from VB6, C, & C++. All other programming languages must use the other standard interfaces.

The custom interfaces are:

- [IArrayTransfer](#) - Measurement object
- [ICalData](#) - Calibrator object
- [ISourcePowerCalData](#) - Channel object

Working with Events

- [What are Events?](#)
- [Using the Analyzer's Events](#)
- [Event ID's](#)
- [Filtering Events](#)
- [List of Events](#)
- [Out of Range Errors](#)
- [Troubleshooting Problems with Events](#)

See Also

[Events Example](#)

[Errors and the SCPIStringParser Object](#)

Other Topics about COM Concepts

What are Events?

Windows applications work from user-initiated events such as mouse moves and mouse clicks. A mouse-click produces an event that the programmer can either ignore or "handle" by providing an appropriate subroutine like this:

```
Sub DoThis_onClick
  Perform something
End Sub
```

If this subroutine were in your program and the mouse-click event occurs on your PC, it would generate a "Callback" to the client and interrupt whatever it was doing and handle the event.

A more practical example of an event in the analyzer is Limit test. If limit test is on and the measurement fails, the analyzer produces a "Limit-failed" event. If the measurement passed, the analyzer produces a "Limit-succeeded" event.

The Analyzer has a very sophisticated Event structure. Your program **CAN** be notified when one or more events occur. However, it may not be necessary.

For example, the analyzer has an event that will notify your program when a sweep is complete. A

simpler alternative is to use a synchronous command which waits for the sweep to complete.

```
sync = True
app.ManualTrigger sync
chan.StartFrequency = 4.5E6
```

This would NOT work if you want the controller to do other things while waiting, like setup a power meter or sort some data. In this case you would like a "callback" from the analyzer to let your program know that the sweep has completed. For an example of this see [Events Example](#).

Another reason to use events is when you want to be notified of several conditions when they occur, such as errors or source unlock conditions. It would not be practical to routinely poll these conditions while executing your program.

Using Events

If you decide to use the COM events to get a callback, your program must do two things:

1. Subscribe to events:

All events in the analyzer are a child of the Application object through the INetworkAnalyzerEvents Interface. You must tell the Application object that you are interested in receiving event callbacks. This process is called subscription.

In Visual Basic, this is done by including "WithEvents" in the declaration statement. The declaration below dimensions an Application object (myPNA) and subscribes to the events produced by the Application.

```
Dim WithEvents myPNA as AgilentPNA835x.Application
```

In C++, this is a bit more involved. You must queryInterface for the IconnectionPointContainer interface, locate the InetworkAnalyzerEvents interface via a call to FindConnectionPoint and call Advise().

2. Implement the Event Handler

When an event occurs, the Application object will "callback" to the client through the InetworkAnalyzerEvents interface.

In VB, click on the object window (upper left pane). Find the Application object and click it. The event interfaces will appear in the upper right pane. As you click on them, VB supplies the first line of code. You fill in the rest of the handler routine to service the event. The following is an example of a event handler subroutine.

Note: In C++, you must type the callback.

List of Events

The following is a list of categories and the general types of events they include. Click the link view the event details.

Category Enumeration	Callback
naEventCategory_PARSER	OnSCPIEvent
naEventCategory_MEASURE	OnMeasurementEvent
naEventCategory_CHANNEL	OnChannelEvent
naEventCategory_HW	OnHardwareEvent
naEventCategory_CAL	OnCalEvent
naEventCategory_USER	OnUserEvent
naEventCategory_DISPLAY	OnDisplayEvent
naEventCategory_GENERAL	OnSystemEvent

Note: Use the [MessageText](#) Method to get a text message describing the event.

Out of Range Errors

When you attempt to set a value on an active function that is beyond the range (min or max) of the allowable values, the analyzer limits that value to an appropriate value (min or max) and sets the function to the limited value. From the front panel controls this is visually evident by the limited value in the edit box or by the annotation on the display. An example would be attempting to set the start frequency below 300kHz. The edit control doesn't allow the number to fall below 300kHz.

When the automation user programs a setting (such as start frequency below the allowable limits) the same behavior takes place. The analyzer accepts the limited value. However, in order to learn what setting took place, you have to read the HRESULT.

All automation calls return HRESULTS. By default the HRESULT returned when an overlimit occurs is S_NA_LIMIT_OUTOFRANGE. This value is a success code, meaning that bit 31 in this 32 value is 0. Programmers should check the return code from all automation calls to determine success or failure.

Some C++ macros (like SUCCEEDED(hr) or FAILED(hr)) only check bit 31. So if you are interested in trapping this outOfRange error you will have to check for S_NA_LIMIT_OUTOFRANGE explicitly.

Alternatively, you can configure the analyzer to report outOfRange conditions with an error code. Use the method: App.[SetFailOnOverRange](#) (true). With this method set TRUE, any overrange error will return E_NA_LIMIT_OUTOFRANGE_ERROR.

This method is provided for the benefit of VB clients. VB users can't detect specific success codes because the VB runtime strips off the HRESULT and only raises a run time error if bit 31 is set,

indicating a fail code.

Troubleshooting Problems with Callbacks

When you do callbacks, the client PC becomes the server and the analyzer (server) becomes the client. Callbacks can only take place when both server and client are in the same workgroup or in the same domain. See [Configure for COM](#).

Read and Write Calibration Data using COM

Calibration data in the PNA is stored in Cal Sets. [Learn more about Cal Sets](#)

You can read or write two types of Calibration data:

- **Error Terms** - calculated data using standard measurement data and the algorithms for the specified cal type.
- **Standard Measurement data** -raw data resulting from the measurement of a calibration standard.

Each of these data are available in the PNA in either variant data or typed data. [Learn more about variant and typed data](#)

Other Topics about COM Concepts

Calibration / Cal Set Interfaces

There are several interfaces associated with Calibration.

[ICalibrator](#)

This interface is the original interface provided with the first version of the PNA. It provides remote access to the "Unguided" Calibration wizard. This interface can perform 1 and 2 port calibrations as well as response cals.

This interface can also read and write error terms from/to a Cal Set. However, ICalibrator is NOT recommended for this purpose. The [ICalSet2](#) Interface is better suited for reading and writing error terms.

See a vbscript example of [how to perform a 2-port Cal and read the cal data.](#)

[IGuidedCalibration](#)

This interface provides the methods and properties used by the Guided Calibration wizard. With this interface you can perform multi-port calibrations (1 to 4 port cals), but no response cals.

[ICalSet2](#) and [ICalData3](#)

These interfaces provide access to the Cal Set contents. You can read and write error terms with both of these interfaces.

- ICalSet2 uses Variant data, which means it is usable from vbscript.

- ICalData3 uses "typed" data, which means it can be used from any automation engine that can read the type library (VEE, VB, C++, etc.). Typed arguments (such as float or single) are more efficient than variants, so use the ICalData3 interface where better performance is needed.

[See a vbscript example of how to read Cal Set data.](#)

ICalSet3

This interface provides access to the stimulus attributes of the Cal data: frequency, power, number of points. These are the stimulus conditions under which the Cal Set was created.

Programming the PNA with C++

The programming information contained in this Help system is aimed at the Visual Basic programmer. VB does a lot of work for the programmer when it comes to managing and accessing components. Using a lower level language like C++ requires a more thorough understanding of the underlying tenets of COM. It is not the intent of this section to teach COM programming. The following is intended to acquaint you with some of the basic concepts you need to know in order to program against COM.

- [Initializing COM](#)
- [Importing the Type Library](#)
- [Creating the Application Object](#)
- [Errors](#)
- [Events](#)
- [Additional Reading](#)
- [Example](#)

Note: The information in this section assumes development on a Windows OS using Microsoft tools.

[Other Topics about COM Concepts](#)

Initializing COM

The first thing you must do before performing any COM transactions is to initialize the COM library. You can do this in a number of ways. The most basic of these is a call to **CoInitialize()** or **CoInitializeEx()**. Alternatively you can use the MFC (Microsoft Foundation Classes) **AfxOleInit()**.

Conversely, before your program exits you must uninitialize COM. You can accomplish this with **CoUninitialize()** or the MFC routine **AfxOleTerm()**.

Importing the Type Library

To make a component available to the client, the server exports what is called the type library. For the PNA, this file is 835x.tlb. It is located on the PNA's hard drive at **C:/Program Files/ Agilent/ Network Analyzer/ Automation**. See [Configure for COM-DCOM Programming](#).

The type library can be read and deciphered using another COM interface called ITypeLib. VB uses this interface to present, for example, its object browser. Visual C++ can also read type libraries. This is done by importing the type library into your project with a compiler directive:

```
#import "C:/Program Files/Common Files/Agilent/Pna/835x.tlb", named_guids
```

When you compile your program with this statement in it, the compiler creates two other files: **835x.tlh** and **835x.tli**. The first is a header file that contains the type definitions for the PNA's COM interfaces and their methods. The second file contains inline functions that wrap the PNA's interface methods. The wrappers are beneficial in that they contain error reporting for each of the method calls.

The .tlh file defines a smart pointer which you can use to access the PNA's objects. The smart pointer definition looks like this:

```
_com_smartptr_typedef(Iapplication, _uuidof(Iapplication))
```

A smart pointer is a term used for a C++ object that encapsulates a pointer used to refer to a COM object. All COM objects derive from the interface IUnknown. This interface has three methods: QueryInterface(), AddRef(), and Release(). The function of the AddRef and Release methods is to maintain a reference count on the object and thus control the object's lifetime. Anytime you copy or create a reference to a COM object, you are responsible for incrementing its reference count. And likewise, when you are finished using that reference, it is your responsibility to Release it. Smart pointers do this work for you, as shown in the [example program](#). In addition, smart pointers will also perform the QueryInterface call when required. QueryInterface is a method that requests a specific interface from an object. In the example program we gain access to the IArrayTransfer interface of the Measurement object. In the ReadMethod routine, we see this:

```
PTransferData = pMeas;
```

The assignment operator is overloaded for the smart pointer and in reality, this simple statement does this:

```
HRESULT hr = pMeas->QueryInterface( IID_IArrayTransfer, (void**)&pTransferData);
```

Using the existing interface pointer (pMeas) to the object, this call asks the object if it supports the IArrayTransfer interface, and if so to return a pointer to it in pTransferData. Smart pointer makes life easier for the C++ programmer. Read more about smart pointers in Microsoft Developer's Network Library (*MSDN*).

Creating the Application Object

The only createable object exported by the PNA is the [Application object](#). Typically this would be done with a call to CoCreateInstance:

```
STDAPI CoCreateInstance(  
    CLSID_IApplication, //Class identifier (CLSID) of the object  
    NULL, //Pointer to controlling IUnknown  
    CLSCTX_SERVER, //Context for running executable code  
    IID_IApplication, //Reference to the IID of the interface  
    (void**)&pNA //Address of output variable that receives
```

```
// the interface pointer requested in riid
);
```

With the smart pointer, this is taken care of with the following call:

```
IApplicationPtr pNA; // declare the smart pointer
pNA = IApplicationPtr("AgilentPNA835x.Application.1");
```

Errors

All COM method calls are required to return an HRESULT. This is 32 bit long with a specific format.

- The most significant bit indicates success(0) or failure(1).
- The lower 16 bits indicate the specific failure.

Visual Basic strips off the returned HRESULT and raises an error object for non-successful returns. The C++ programmer must himself be diligent about handling errors. You must check the return value of each COM call to ensure its success.

Events

The Application object sources the INetworkAnalyzerEvents interface. This object is the source for all events. To use events in C++, you must do two things:

1. Implement the INetworkAnalyzerEvents interface - derive an object from INetworkAnalyzerEvents and implement the methods described there.
2. Subscribe to the IconnectionPoint interface of the Application object. - obtain a pointer to the IconnectionPointContainer interface of the Application object and making the following request:

```
FindConnectionPoint( IID_InetworkAnalyzerEvents, &pConnection );
```

A successful call to this interface will return a valid pointer in pConnection. Use this pointer to subscribe to the Application object:

```
pConnect->Advise( IUnknown* punk, DWORD dwCookie);
```

This call provides the server object with a callback address. The Iunkown pointer in this call is the IUnkown pointer of the object that implements the INetworkAnalyzerEvents interface. This is the event sink. The application object needs a pointer to this object in order to call your interface when an event occurs. The **dwCookie** is your subscription key. Use it to unsubscribe (see Unadvise()).

Additional Reading

"MSDN" - Microsoft Developer's Network Library

"Learning DCOM", by Thuan L. Thai, published by O'Reilly(1999)

"Inside COM", by Dale Rogerson, published by Microsoft Press (1997)

"Understanding ActiveX and OLE", by David Chappell, also published by Microsoft Press (1996)

"Beginning ATL COM Programming", published by Wrox Press (1998)

Example

The example uses the smart pointer created by Microsoft Visual Studio. The calls to CoInitialize and CoUninitialize open and close the COM libraries. In the example, notice that the pointers local to the main routine are explicitly released. When smart pointers go out of scope, they will perform this duty implicitly. However, we are calling CoUninitialize before they have the chance to be destroyed, so we are obliged to release them.

See the [example](#) program.

Using COM from .NET

To communicate with the PNA from Microsoft .NET enabled languages such as C# and Visual Basic.NET perform the following steps:

1. [Configure your PC and PNA for COM-DCOM Programming.](#)
2. Reference the type library within the development environment (see the following [exception for managed C++ projects.](#)) In the process of referencing the type library, a .NET assembly is created that wraps the PNA type library with a .NET friendly interface. This .NET assembly is called an Interop Assembly.

Note: ONLY 32-bit compiler option is supported (64-bit is NOT supported).

Exception for managed C++ projects: To generate the Interop Assembly for managed C++ projects, you must use the `tlbimp.exe` utility. This utility is described in the MSDN documentation. On your PC, click Start then Run then type: `tlbimp.exe 835x.tlb` and click OK. After doing this you can use the `#using` directive to include the Interop Assembly on managed C++ projects.

Example: Creating a .NET object from C#

The following is an example that shows how to create a .NET object that connects to the PNA over DCOM. In this example, *machineName* is either the DNS name or the IP address of the PNA to connect with.

```
Type pna = Type.GetTypeFromProgID("AgilentPNA835x.Application", machineName);  
  
AgilentPNA835x.IApplication app =  
(AgilentPNA835x.IApplication)Activator.CreateInstance(pna);
```

See C# Example Programs:

[Perform a Guided Cal with C#](#)

[Using C#](#)

Registering the PNA Primary Interop Assembly (PIA) (OPTIONAL)

The PIA is NOT necessary to communicate with the PNA. The following procedure is useful only when there are two .NET programs that want to share the same PNA interface definitions. Without the PIA, each .NET application would use its own Interop Assembly.

To register the PIA on a machine, you need to have the common language runtime (CLR) installed. This is included with Visual Studio.NET. Then perform the following steps:

Note: In the following steps, replace `<local directory>` with the full path name of the specified file on your PC.

1. Run the PNAProxy.exe program as described in [Configure for COM-DCOM Programming](#).
2. On the PNA, copy **C:/Program Files/Agilent/Network Analyzer/Automation/AgilentPNA835x.dll** to a local directory on your PC. Make a note of this directory.
3. On your PC, click **Start**, then **Run**, then type: **regasm <local directory> /AgilentPNA835x.dll** and click **OK** to register the dll.
4. Again, click **Start**, then **Run**, then type: **gacutil /i <local directory> /AgilentPNA835x.dll** and click **OK** to add the assembly to the Global Assembly Cache (GAC).

To **Uninstall the PIA**, perform the following:

1. On your PC, click **Start**, then **Run**, then type: **gacutil /u <local directory> /AgilentPNA835x.exe** and click **OK** to remove the assembly from the GAC.
2. On your PC, click **Start**, then **Run**, then type: **regasm /unregsite <local directory> / agilentpna835x.dll** and click **OK** to unregister the assembly.
3. To uninstall PNA Proxy.exe use the **Add/Remove Programs** utility in the control panel.

Last Modified:

2-Feb-2009 Updated by JE

SCPI Command Tree

See Also

- [Example Programs](#)
 - [Find commands using a simulated PNA UI](#)
 - [See list of all SCPI Errors.](#)
 - [See Calibrating the PNA Using SCPI](#)
 - [Synchronizing the PNA and Controller](#)
 - [IEEE- 488.2 Common Commands](#)
 - [Local Lockout](#)
-

ABORt	Stops all sweeps
+ CALCulate	Click to hide and show CALC branches
CORRection	Electrical Delay and Phase Offset
CUSTom	Custom measurements
DATA	Sends and queries data.
EQUation	Equation Editor
FILTer	Time domain gating
FORMat	Display format
FSIMulator	Balanced measurements and Fixturing
FUNction	Trace Statistics
GCData	Read Gain compression data
GCMeas	Gain Compression Analysis
GDELay	Group Delay Aperture setting
LIMit	Limit lines for pass / fail testing
MARKer	Marker settings
MATH	Math / Memory
MIXer	X-axis display for FCA measurements
NORMALize	Receiver power cal (Obsolete)
OFFSet	Mag and Phase offset
PARAmeter	Create and delete measurements
RDATA?	Queries receiver data
SMOothing	Point-to-point smoothing
TRANSform	Time domain transform

X:VALues	Returns X-Axis values for trace
CALPod	Controls CalPod units
CONTRol	Interface control, ECal module state control, and Rear-panel connector control.
CSET	Work with a Cal Set without having to select it into that channel.
DISPlay	Display settings
FORMat	Format for data transfer
HCOPY	Hardcopy printing
INITiate	Continuous or manual triggering
LXI	LXI communications
MMEMory	Saves and recalls instrument states
OUTPut	Turns RF power ON and OFF
ROUTe	Controls internal switch to reference receiver. (Opt 81)

+ SENSE Click to hide and show SENSE branches

AVERage	Sweep Averaging
BANDwidth	IF Bandwidth
CLASs	Returns measurement class name
CORRection	Calibration and other correction settings
CKIT	Manage Cal Kits and ECal modules
COLL:CKIT	Edit Cal Kit definitions
COLL:GUIDed	Perform Guided Cals
PSEnsor	Configure Guided Power Cal
SMC	Perform SMC Cal
VMC	Perform VMC Cal
CSET	Manage Cal Sets
EXTension	Port Extensions
COUPle	Chopped or Alternate sweep
FOM	Frequency Offset (opt 080)
FREQuency	Frequency sweep settings
GCSetup	Gain Compression App (opt 086)
IF	IF Access settings
IMD	Intermodulation Distortion (opt 087)
IMS	Intermodulation Spectrum (opt 087)
MIXer	FCA measurements (opts 082 and 083)
MULTiplexer	Controls external test sets.
NOISe	Noise Figure (opt 029)

PATH	Provides access to hardware configuration
POWer	Receiver attenuation and overpower protection
PULSe	Configure internal pulse generators
ROLE	Assign sources to roles.
ROSCillator	Returns the source of the reference oscillator.
SEGment	Segment sweep settings.
SWEep	Sweep types
TEMPerature	Returns the temperature on the receiver board
X:VALues	Returns X-axis values for channel
SOURce	Source power to the DUT
DC	DC Source control
PHASe	Phase control (Opt 088)
Power:CORR	Source power Calibration
ALC:MODE:REC	Receiver Leveling
STATus	Reads the PNA status registers
SYSTEM	Misc PNA capabilities
CAL:All	Calibrate All Channels
CAL:Phase	Perform an SMC Phase Reference Cal
CONF:EDEV	Configure external devices
PREFerences	PNA Preferences
TRIGger	Trigger measurements

Last Modified:

22-Sep-2011	Added Temp (A.09.50)
5-Jan-2011	Added Phase control
13-Apr-2010	Added CSET (A.09.20)
23-Feb-2010	Added CALC:GDElay and LXI (A.09.20)
30-Jul-2009	Added syst:conf:edev and Calc:X?
16-Jan-2009	Moved class

IEEE 488.2 Common Commands

[*CLS - Clear Status](#)

[*ESE - Event Status Enable](#)

[*ESE? - Event Status Enable Query](#)

[*ESR? - Event Status Enable Register](#)

[*IDN? - Identify](#)

[*OPC - Operation complete command](#)

[*OPC? - Operation complete query](#)

[*OPT? - Identify Options Query](#)

[*RST - Reset](#)

[*SRE - Service Request Enable](#)

[*SRE? - Service Request Enable Query](#)

[*STB? - Status Byte Query](#)

[*TST? - Result of Self-test Query](#)

[*WAI - Wait](#)

See Also

- [Example Programs](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

***CLS - Clear Status**

Clears the instrument status byte by emptying the error queue and clearing all event registers. Also cancels any preceding *OPC command or query. See [Status Commands](#) and [Reading the Analyzer's Status Registers](#).

***ESE - Event Status Enable**

Sets bits in the standard event status enable register. See [Status Commands](#) and [Reading the Analyzer's Status Registers](#).

***ESE? - Event Status Enable Query**

Returns the results of the standard event enable register. The register is cleared after reading it. See [Status Commands](#) and [Reading the Analyzer's Status Registers](#).

***ESR - Event Status Enable Register**

Reads and clears event status enable register. See [Status Commands](#) and [Reading the Analyzer's Status Registers](#).

***IDN? - Identify**

Returns a string that uniquely identifies the analyzer. The string is of the form "Agilent Technologies",<model number>,<serial "number">,<software revision>" .

Note: Beginning with Rev 6.01, this command now returns the software revision with 6 digits instead of 4. For example, A.06.01.02.

***OPC - Operation complete command**

Generates the OPC message in the standard event status register when all pending overlapped operations have been completed (for example, a sweep, or a Default). See [Understanding Command Synchronization](#).

***OPC? - Operation complete query**

Returns an ASCII "+1" when all pending overlapped operations have been completed. See [Understanding Command Synchronization](#)

***OPT? - Identify Options Query**

Returns a string identifying the analyzer option configuration.

***RST - Reset**

Executes a device reset and cancels any pending *OPC command or query, exactly the same as a [SYSTem:PRESet](#) with one exception: Syst:Preset does NOT reset [Calc:FORMAT](#) to ASCII. The contents of the analyzer's non-volatile memory are not affected by this command.

***SRE - Service Request Enable**

Before reading a status register, bits must be enabled. This command enables bits in the service request register. The current setting is saved in non-volatile memory. See [Status Commands](#) and [Reading the Analyzer's Status Registers](#).

***SRE? - Service Request Enable Query**

Reads the current state of the service request enable register. The register is cleared after reading it. The return value can be decoded using the table in [Status Commands](#). See also [Reading the Analyzer's](#)

Status Registers.

***STB? - Status Byte Query**

Reads the value of the instrument status byte. The register is cleared only when the registers feeding it are cleared. See [Status Commands](#) and [Reading the Analyzer's Status Registers](#).

***TST? - Result of Self-test Query**

Returns the result of a query of the analyzer hardware status. An **0** indicates no failures found. Any other value indicates one or more of the following conditions exist. The value returned is the Weight (or sum of the Weights) of the existing conditions. For example:

- If **4** is returned from *TST?, an **Overpower** condition exists.
- If **6** is returned, both **Unleveled** and **Overpower** conditions exist.

Bit	Weight	Description	Bit is set to 1 when the following conditions exist:
0	1	Phase Unlock	The source has lost phaselock. This could be caused by a reference channel open or a hardware failure.
1	2	Unleveled	The source power is unleveled. This could be a source is set for more power than it can deliver at the tuned frequency. Or it could be caused by a hardware failure.
2	4	Not used	
3	8	EE Write Failed	An attempted write to the EEPROM has failed. This is possibly caused by a hardware failure.
4	16	YIG Cal Failed	The analyzer was unable to calibrate the YIG. Either the phaselock has been lost or there has been a hardware failure.
5	32	Ramp Cal Failed	The analyzer was unable to calibrate the analog ramp generator due to a possible hardware failure.
6	64	Not used	

***WAI - Wait**

Prohibits the instrument from executing any new commands until all pending overlapped commands have been completed. See [Understanding Command Synchronization](#)

Last Modified:

17-Sep-2008 Added *RST vs Syst:Pres note

Abort Command

ABORt

(Write-only) Stops all sweeps - then resume per current trigger settings. This command is the same as [INITtiate:IMMEDIATE](#) (restart) except if a channel is performing a single sweep, ABORt will stop the sweep, but not initiate another sweep.

Learn about [Synchronizing the PNA and Controller](#)

Examples	ABOR abort
Query Syntax	Not applicable
Default	Not applicable

Calculate:Correction Commands

Controls error correction functions.

CALCulate:CORRection
EDELay
DISTance
TIME
MEDium
UNIT
WGCutoff
[STATe]
INDicator?
TYPE
OFFSet
[MAGNitude]
PHASe

Click on a [blue](#) keyword to view the command details.

[Red](#) keywords are superseded.

See Also

- [Example Programs](#)
- [Calibrating the PNA Using SCPI](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using [Calc:Par:MNUM](#) or [Calc:Par:Select](#). [Learn more](#).

CALCulate<cnum>:CORRection:EDELay:DISTance <num>

(Read-Write) Sets the electrical delay in physical length (distance) for the selected measurement.

[See Critical Note](#)

Parameters

<cnun> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnun> is set to 1.

<num> Electrical delay in distance.

First Specify units using [CALC:CORR:EDEL:UNIT](#)

Use [SENS:CORR:RVEL:COAX](#) <num> to set Velocity factor.

This parameter supports MIN and MAX as arguments. [Learn more.](#)

Examples

```
CALC1:CORR:EDEL:DIST 5
```

```
calculate2:correction:distance .003
```

Query Syntax CALCulate:CORRection:EDELay:DISTance?

Return Type Numeric

Default 0

CALCulate<cnun>:CORRection:EDELay:MEDium <char>

(Read-Write) Sets the media used when calculating the electrical delay.

Parameters

<cnun> Any existing channel number. If unspecified, value is set to 1.

<num> Choose from: **COAX** for coaxial medium, **WAVE**guide for waveguide medium.

Examples

```
CALC:CORR:EDEL:MED COAX
```

```
calc3:corr:edelay:medium waveguide
```

Query Syntax CALCulate<cnun>:CORRection:EDELay:MEDium?

Return Type Character

Default COAX

CALCulate<cnun>:CORRection:EDELay:UNIT <char>

(Read-Write) Sets and returns the units for specifying electrical delay in physical length (distance).

Parameters

<num> Any existing channel number. If unspecified, value is set to 1.

<char> Units for delay in distance. Choose from:

- METer
- FEET
- INCH

Examples

```
CALC:CORR:EDEL:UNIT MET  
calc3:corr:edelay:unit inch
```

Query Syntax CALCulate<num>:CORRection:EDELay:UNIT?

Return Type Character

Default METer

CALCulate<num>:CORRection:EDELay[:TIME] <num>

(Read-Write) Sets the electrical delay for the selected measurement.

[See Critical Note](#)

Parameters

<num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

<num> Electrical delay in seconds. Choose any number between:

-10.00 and 10.00

Use [SENS:CORR:RVEL:COAX](#) <num> to set Velocity factor.

This parameter supports MIN and MAX as arguments. [Learn more.](#)

Examples

```
CALC1:CORR:EDEL:TIME 1NS  
calculate2:correction:time 0.5e-12
```

Query Syntax CALCulate:CORRection:EDELay[:TIME]?

Return Type Numeric

Default 0 seconds

CALCulate<cnum>:CORRection:EDELay:WGCutoff <num>

(Read-Write) Sets the waveguide cutoff frequency used when the electrical delay media is set to WAVEguide. (See [CALCulate:CORRection:EDELay:MEDIum <char>](#).)

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1.
- <num> Waveguide cutoff frequency used with the electrical delay calculation.

This parameter supports MIN and MAX as arguments. [Learn more.](#)

Examples

```
CALC:CORR:EDEL:WGC 18.067 GHz  
calculate3:correction:edelay:wgcutoff 14.047 ghz
```

Query Syntax CALCulate<cnum>:CORRection:EDELay:WGCutoff?

Return Type Numeric

Default 45 MHz

CALCulate<cnum>:CORRection[:STATe] <bool>

(Read-Write) Turns error correction ON or OFF for the selected measurement on the specified channel.

To turn error correction ON or OFF for a channel, use [SENS:CORR:STATe](#).

[See Critical Note](#)

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <bool> Correction state. Choose from:

0 - Correction OFF

1 - Correction ON

Examples

```
CALC:CORR ON  
calculate:correction:state off
```

Query Syntax CALCulate<cnum>:CORRection:STATe?

Return Type Boolean

Default Not Applicable

CALCulate<cnum>:CORRection[:STATe]:INDicator?

(Read-only) Returns the error correction state for the selected measurement on the specified channel.

To turn error correction ON or OFF for a channel, use [SENS:CORR:STATe](#).

[See Critical Note](#)

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

Examples

```
CALC:CORR:IND?
```

```
calculate2:correction:state:indicator?
```

Return Type

Character

NONE - No error correction

MAST (Master) - Original error correction terms

INT - Error terms are interpolated. [Learn more](#).

DELT - Delta Match calibration terms. [Learn more](#).

INV - Error terms are not valid

Default NONE

CALCulate<cnum>:CORRection:TYPE <string>

(Read-Write) Sets the Cal Type for the selected measurement on the specified channel. This is used when a Cal Set is applied. [Learn more about applying Cal Types](#).

- Use [SENS:CORR:TYPE:CAT?](#) to list the Cal Types in the PNA.
- Use [SENS:CORR:CSET:TYPE:CAT?](#) to list the Cal Types contained in the active Cal Set for the channel.
- Use [SENS:CORR:COLL:METH](#) to set the Cal type to perform a new Unguided calibration,

[See Critical Note](#)

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<string> (**String**) Cal type. Case sensitive. Use one of the following:

For Full Calibrations (NO Power Cal included):

This command does not distinguish between TRL and SOLT. The same number of error terms is applied for both Cal Types.

```
"Full <n> Port(x,y,z...)"
```

where

<n> = the number of ports to calibrate

x,y,z = the port numbers to calibrate

For example:

```
"Full 4 Port(1,2,3,4)"
```

For Full Calibrations (including Power Cal):

After the Full <n> port, include the string, "with power"

For example:

```
"Full 4 Port with power(1,2,3,4)"
```

For Response Calibrations:

```
"Response(param)" OR
```

```
"ResponseAndIsolation(param)"
```

Where param =

- S-parameter. For example"
 - ```
"Response(S21)"
```
  - ```
"ResponseAndIsolation(A/R)"
```
- Single or ratioed receivers using either [logical receiver notation](#) or physical receiver notation. For example:
 - ```
"Response(A)"
```
  - ```
"ResponseAndIsolation(a3/b4)"
```

For Enhanced Response Calibrations:

```
"EnhancedResp(sourcePort, recPort)"
```

Where:

- sourcePort = stimulus port number
- recPort = receiver port number

For FCA Calibrations:

[Learn more about this setting.](#)

- "SMC_2P" (Response + Input + Output) All four sweeps required. Most accurate.
- "SMCRsp+IN" No Output match. All four sweeps required.
- "SMCRsp+OUT" No Output match. All four sweeps required.
- "SMCRsp" No Input or Output match. Saves two sweeps.

For VMC, multiple Cal types are not available.

For Gain Compression Cal

where r = receive port; s = source port

- "GCA 2P (r,s)" - full 2-port cal
- "GCA Enh Resp (r,s)" - Enhanced Response Cal

Examples

```
CALC:CORR:TYPE "Scalar Mixer Cal"
```

Query Syntax

```
CALCulate<num>:CORRection:TYPE?
```

Return Type

String

Default

Not Applicable

CALCulate<num>:CORRection:OFFSet[:MAGNitude] <num> **Superseded**

Note: This command is replaced with [SENS:CORR:RPOWER:OFFSet\[:AMPLitude\]](#).
To set data trace magnitude offset, use [CALC:OFFS:MAGN](#)
This command does NOT function for FCA measurements.

See an example of a [Receiver Power Calibration](#).

(Read-Write)

For Receiver Power Calibration, specifies the power level to which the selected (unratioed) measurement data is to be adjusted. This command applies only when the selected measurement is of unratioed power.

Parameters

- <num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.
- <num> Cal power level in dBm. No limits are enforced on this value, but the PNA receivers themselves have maximum and minimum power specifications (that may differ between PNA models) which this value must comply with for a valid receiver power cal.

Examples

```
CALC:CORR:OFFS 10DBM  
calculatel:correction:offset:magnitude maximum
```

Query Syntax CALCulate<num>:CORRection:OFFSet[:MAGNitude]?

Return Type Numeric

Default 0dBm

CALCulate<num>:CORRection:OFFSet:PHASe <num>[<char>] **Superseded**

Note: This command is replaced with [CALC:OFFS:PHASe](#)

(Read-Write) Sets the phase offset for the selected measurement.

Parameters

- <num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.
- <num> Offset phase value. Choose any number between:
-360 and **360**
- <char> Units for phase. OPTIONAL. Choose either:
DEG - Degrees (default)
RAD - Radians

Examples

```
CALC:CORR:OFFS:PHAS 10  
calculate:correction:offset:phase 20rad
```

Query Syntax CALCulate:CORRection:OFFSet:PHASe?

Return Type Numeric, returned value always in degrees

Default 0 degrees

Last modified:

19-Jul-2010 Added Calc:Corr:Indicator
22-Sep-2009 Removed VMC from Corr:Type
6-Feb-2009 Added two commands
12-Feb-2008 Fixed typo
9/12/06 MQ Modified Calc:Corr for multiport.

Calculate:Custom Commands

Creates and modifies application measurements:

CALCulate:CUSTom:

DEFine

MODify

See Also

- [Example Programs](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

CALCulate<cnum>:CUSTom:DEFine <Mname>, <type> [,param]

(Write-only) Creates a custom measurement. The custom measurement is not automatically displayed. You must also do the following:

- Use [DISP:WIND:STATe](#) to create a window if it doesn't already exist.
- Use [DISP:WIND:TRAC:FEED](#) to display the measurement
- Select the measurement ([CALC:PAR:SEL](#)) before making additional settings.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1.

<Mname> Name of the measurement. Any non-empty, unique string, enclosed in quotes.

<type> String. The type of custom measurement. Click each to see an example program. Choose from:

- ["Vector Mixer/Converter"](#)
- ["Scalar Mixer/Converter"](#)
- ["Noise Figure Cold Source"](#)
- ["Noise Figure Converters"](#)
- ["Gain Compression"](#)
- ["Gain Compression Converters"](#)

- ["Swept IMD"](#)
- "IM Spectrum"
- ["Swept IMD Converters"](#)
- "IM Spectrum Converters"

[param] String. Optional parameter specifies the measurement to create:

Meas Class	Measurement Name	Description
Vector Mixer/Converter	"S11"	Learn about VMC parameters
	"VC21"	
	"S22"	
Scalar Mixer/Converter	"S11"	Learn about SMC parameters
	"SC21"	
	"SC12"	
	"S22"	
	"Ipwr"	
	"RevIPwr"	
	"Opwr"	
Gain Compression Learn more	GCA and GCX:	
	"CompIn21"	Input power at the compression point.
	"CompOut21"	Output power at the compression point.
	"CompGain21"	Gain at the compression point.
	"CompS11"	Input Match at the compression point
	"RefS21"	Linear Gain
	"DeltaGain21"	CompGain21 -Linear Gain
	"S11", "S21", "S12", "S22"	Standard S-parameters; measured at port 1 and port 2

<p>Gain Compression Converters</p> <p>Learn more</p>	<p>GCX - All Gain Compression parameters (except S21 and S12) plus the following:</p> <p>"S11"</p> <p>"SC21"</p> <p>"SC12"</p> <p>"S22"</p> <p>"Ipwr" Mixer parameters</p> <p>"RevIPwr"</p> <p>"Opwr"</p> <p>"RevOPwr"</p>																										
<p>Noise Figure Cold Source</p>	<p>Noise Figure AND NFX:</p> <table border="1"> <tr> <td>"NF"</td> <td>Noise figure</td> </tr> <tr> <td>"ENR"</td> <td>Validate noise source measurements.</td> </tr> <tr> <td>"T-Eff"</td> <td>Effective noise temperature.</td> </tr> <tr> <td>"DUTRNP"</td> <td>DUT noise power ratio. (Noise power expressed in Kelvin divided by 290).</td> </tr> <tr> <td>"DUTRNPI"</td> <td></td> </tr> <tr> <td>"SYSRNP"</td> <td>System noise power ratio</td> </tr> <tr> <td>"SYSRNPI"</td> <td></td> </tr> <tr> <td>"DUTNPD"</td> <td>DUT noise power density. (Noise power expressed in dBm/Hz).</td> </tr> <tr> <td>"DUTNPDI"</td> <td></td> </tr> <tr> <td>"SYSNPD"</td> <td>System noise power density.</td> </tr> <tr> <td>"SYSNPDI"</td> <td></td> </tr> <tr> <td>"OvrRng" (Opt 029 Only)</td> <td>Indication that the noise receiver is being over powered.</td> </tr> <tr> <td>"T-Rcvr" (Opt 029 Only)</td> <td>Temperature reading (in Kelvin) of the noise receiver board.</td> </tr> </table>	"NF"	Noise figure	"ENR"	Validate noise source measurements.	"T-Eff"	Effective noise temperature.	"DUTRNP"	DUT noise power ratio. (Noise power expressed in Kelvin divided by 290).	"DUTRNPI"		"SYSRNP"	System noise power ratio	"SYSRNPI"		"DUTNPD"	DUT noise power density. (Noise power expressed in dBm/Hz).	"DUTNPDI"		"SYSNPD"	System noise power density.	"SYSNPDI"		"OvrRng" (Opt 029 Only)	Indication that the noise receiver is being over powered.	"T-Rcvr" (Opt 029 Only)	Temperature reading (in Kelvin) of the noise receiver board.
"NF"	Noise figure																										
"ENR"	Validate noise source measurements.																										
"T-Eff"	Effective noise temperature.																										
"DUTRNP"	DUT noise power ratio. (Noise power expressed in Kelvin divided by 290).																										
"DUTRNPI"																											
"SYSRNP"	System noise power ratio																										
"SYSRNPI"																											
"DUTNPD"	DUT noise power density. (Noise power expressed in dBm/Hz).																										
"DUTNPDI"																											
"SYSNPD"	System noise power density.																										
"SYSNPDI"																											
"OvrRng" (Opt 029 Only)	Indication that the noise receiver is being over powered.																										
"T-Rcvr" (Opt 029 Only)	Temperature reading (in Kelvin) of the noise receiver board.																										

<p>Learn more</p> <p>Noise Figure Converters</p> <p>Learn more</p>	<p>Noise Figure ONLY - NOT NFX:</p> <table border="1"> <tr> <td data-bbox="776 170 1104 380"> <p>"S11", "S21", "S12", "S22"</p> </td> <td data-bbox="1104 170 1476 380"> <p>Standard S-parameters; measured with the port1 and port2 noise switches set for noise mode.</p> </td> </tr> <tr> <td data-bbox="776 380 1104 506"> <p>"A_1", "A_2" ...and so forth.</p> </td> <td data-bbox="1104 380 1476 506"> <p>Unratioed parameters; with notation: "receiver, source port"</p> </td> </tr> </table> <p>NFX ONLY:</p> <table border="1"> <tr> <td data-bbox="776 506 1104 1087"> <p>"S11"</p> <p>"SC21"</p> <p>"SC12"</p> <p>"S22"</p> <p>"Ipwr"</p> <p>"RevIPwr"</p> <p>"Opwr"</p> <p>"RevOPwr"</p> </td> <td data-bbox="1104 506 1476 1087"> <p>Mixer parameters</p> </td> </tr> <tr> <td data-bbox="776 1087 1104 1178"> <p>"ALO1", " BLO1" ...and so forth.</p> </td> <td data-bbox="1104 1087 1476 1178"> <p>Test port receiver at LO1 frequency</p> </td> </tr> <tr> <td data-bbox="776 1178 1104 1333"> <p>"R1_1", "B_2" ..and so forth.</p> </td> <td data-bbox="1104 1178 1476 1333"> <p>Unratioed parameters with notation: "receiver_source port"</p> </td> </tr> </table>	<p>"S11", "S21", "S12", "S22"</p>	<p>Standard S-parameters; measured with the port1 and port2 noise switches set for noise mode.</p>	<p>"A_1", "A_2" ...and so forth.</p>	<p>Unratioed parameters; with notation: "receiver, source port"</p>	<p>"S11"</p> <p>"SC21"</p> <p>"SC12"</p> <p>"S22"</p> <p>"Ipwr"</p> <p>"RevIPwr"</p> <p>"Opwr"</p> <p>"RevOPwr"</p>	<p>Mixer parameters</p>	<p>"ALO1", " BLO1" ...and so forth.</p>	<p>Test port receiver at LO1 frequency</p>	<p>"R1_1", "B_2" ..and so forth.</p>	<p>Unratioed parameters with notation: "receiver_source port"</p>
<p>"S11", "S21", "S12", "S22"</p>	<p>Standard S-parameters; measured with the port1 and port2 noise switches set for noise mode.</p>										
<p>"A_1", "A_2" ...and so forth.</p>	<p>Unratioed parameters; with notation: "receiver, source port"</p>										
<p>"S11"</p> <p>"SC21"</p> <p>"SC12"</p> <p>"S22"</p> <p>"Ipwr"</p> <p>"RevIPwr"</p> <p>"Opwr"</p> <p>"RevOPwr"</p>	<p>Mixer parameters</p>										
<p>"ALO1", " BLO1" ...and so forth.</p>	<p>Test port receiver at LO1 frequency</p>										
<p>"R1_1", "B_2" ..and so forth.</p>	<p>Unratioed parameters with notation: "receiver_source port"</p>										
<p>Swept IMD</p> <p>Swent IMD</p>	<p>There are over 150 possible Swept IMD parameters, too many to list here.</p> <p>Build the parameters with the Swept IMD Parameter dialog, then copy the parameter name to the remote command.</p> <p>The following are a few example parameters:</p> <table border="1"> <tr> <td data-bbox="776 1654 1104 1772"> <p>"PwrMainLo"</p> </td> <td data-bbox="1104 1654 1476 1772"> <p>Absolute power of the Low tone at the DUT output.</p> </td> </tr> <tr> <td data-bbox="776 1772 1104 1814"></td> <td data-bbox="1104 1772 1476 1814"> <p>Power of the third</p> </td> </tr> </table>	<p>"PwrMainLo"</p>	<p>Absolute power of the Low tone at the DUT output.</p>		<p>Power of the third</p>						
<p>"PwrMainLo"</p>	<p>Absolute power of the Low tone at the DUT output.</p>										
	<p>Power of the third</p>										

IM Spectrum Converters Learn more	"IM3"	product relative to the average power of the f1 and f2 tones measured at the DUT output.
	"OIP3"	Theoretical power level at which the third product will be the same power level as the average of the main tones at the output of the DUT.
IM Spectrum Learn more	"Output"	View signals OUT of the DUT and into PNA port 2 (B receiver).
	"Input"	View signals IN to the DUT (R1 receiver).
	"Reflection"	View signals reflected off the DUT input and back into PNA port 1 (A receiver)
IMx Spectrum Converters Learn more	"Output"	View signals OUT of the DUT and into PNA port 2 (B receiver)

Examples

```
CALC:CUST:DEF 'My VC21', 'Vector Mixer/Converter', 'S22'
calculate2:custom:define 'MyNF', 'NoiseFigure', 'NF'
```

Query Syntax Not applicable

[Overlapped?](#) No

Default Not applicable

CALCulate<cnum>:CUSTom:MODify <param>

(Write-only) Changes the selected custom measurement to a different parameter.

See an example using this command for a [VMC](#) and [SMC](#) measurement

Parameters

- <cnm> Channel of the custom measurement to be changed. First, select the measurement using [CALC:PAR:SEL](#).
- <param> Parameter to change the custom measurement to. Select a parameter that is valid for the type of measurement. Choose from the same arguments as [Calc:Cust:Def](#).

Examples

```
SYST:PRES  
CALC2:CUST:DEF 'My VC21', 'Vector Mixer/Converter'  
CALC:PAR:SEL 'My VC21'  
CALC2:CUST:MOD 'S22'
```

Query Syntax Not applicable

[Overlapped?](#) No

Default Not applicable

Last Modified:

14-Sep-2012	Fixed unratiod notation
27-Sep-2011	Added (Opt 029 Only)
2-Mar-2010	Fixed IM spectrum converters
27-Feb-2009	Added IMD Converters and Noise
8-Sep-2008	Added IMD arguments
23-Aug-2007	Added Noise and GC Arguments

Calculate:Data Commands

Controls writing and reading PNA measurement data.

CALCulate:DATA
CUSTom
CATalog?
SNP?
PORTs?
SAVE

Click on a [blue](#) keyword to view the command details.

Red is a superseded command.

See Also

- [Example Programs](#)
- [Data Access Map](#)
- [Synchronizing the PNA and Controller](#)
- To read receiver data, use [CALC:RDATA?](#)
- To read error terms, use [SENS:CORR:CSET:DATA](#)
- To read SnP measurement data, use [CALC:DATA:SNP?](#)
- [SCPI Command Tree](#)

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using [Calc:Par:MNUM](#) or [Calc:Par>Select](#). [Learn more](#).

(Write) **CALCulate<num>:DATA <char>,<data>**

(Read) **CALCulate<num>:DATA? <char>**

Reads or writes Measurement data, Memory data, or Normalization Divisor data from the [Data Access Map](#) location.

- For Measurement data, use FDATA or SDATA
- For Memory data, use FMEM or SMEM. When querying memory, you must first store a trace into memory

using [CALC:MATH:MEMorize](#).

- For Normalization Divisor (Receiver Power Cal error term) data, use SDIV
- Use [FORMat:DATA](#) to change the data type (<REAL,32>, <REAL,64> or <ASCIi,0>).
- Use [FORMat:BORDER](#) to change the byte order. Use "NORMal" when transferring a binary block from LabView or Vee. For other programming languages, you may need to "SWAP" the byte order.

[Equation Editor](#) Notes:

- When equation editor is active on a trace in a standard S-parameter channel, Calc:Data returns the data from the parameter on the trace that was measured last. For example, for the equation "S22 + S33 + S11", then S33 is the last measured parameter because it uses source port 3.
- In [applications](#), if equation editor is active and the original parameter for the trace is not requested anywhere in the channel, then zeros are returned. If the original parameter is being measured within the channel, then data for the original parameter is returned.
- In general, if an equation contains no measurement parameters, then data for the original parameter is returned.

Note: The Calc:Data SCORR command to read / write error terms is **Superseded** with [SENS:CORR:CSET:DATA](#). SCORR commands do NOT accommodate greater than 12 error terms.

[See Critical Note](#)

Parameters

<num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

<char> **FDATA** Formatted **measurement** data to or from [Data Access Map](#) location **Display** (access point 2).

- Corrected data is returned when correction is ON.
- Uncorrected data is returned when correction is OFF.
- Returns TWO numbers per data point for Polar and Smith Chart format.
- Returns one number per data point for all other formats.
- Format of the read data is same as the displayed format.

SDATA Complex measurement data.

Writes data to [Data Access Map](#) location **Raw Measurement** (access point 0).

- When writing corrected data, and correction is ON, it will be corrected again, resulting in meaningless data.

Reads data from **Apply Error Terms** (access point 1).

- Returns TWO numbers per data point.
- Corrected data is returned when correction is ON.
- Uncorrected data is returned when correction is OFF.

FMEM Formatted memory data to or from [Data Access Map](#) location **Memory result** (access point 4).

- Returns TWO numbers per data point for Polar and Smith Chart format.
- Returns one number per data point for all other formats.
- Format of the read data is same as the displayed format.
- Returned data reflects the correction level (On|OFF) when the data was stored into memory.

SMEM Complex measurement data to or from [Data Access Map](#) location **Memory** (access point 3).

- Returns TWO numbers per data point.
- Returned data reflects the correction level (On|OFF) when the data was stored into memory.
- Returned data reflects the correction level (On|OFF) when the data was stored into memory.

SDIV Complex data from [Data Access Map](#) location **Normalization (5)**.

- Returns TWO numbers per data point.
- If normalization interpolation is ON and the number of points changes after the initial normalization, the divisor data will then be interpolated.
- When querying the normalization divisor, you must first store a divisor trace using [CALC:NORMAlize\[:IMMediate\]](#).

The following Calc:Data SCORR command to read / write error terms is **Superseded** with [SENS:CORR:CSET:DATA](#). These SCORR commands do NOT accommodate greater than 12 error terms.

For 2-Port SOLT and TRL calibrations	Specify this <char>	to get or put this Error Term...
	SCORR1	Forward Directivity
	SCORR2	Forward Source Match
	SCORR3	Forward Reflection Tracking
	SCORR4	Forward Isolation
	SCORR5	Forward Load Match
	SCORR6	Forward Transmission Tracking
	SCORR7	Reverse Directivity
	SCORR8	Reverse Source Match
	SCORR9	Reverse Reflection Tracking
	SCORR10	Reverse Isolation
	SCORR11	Reverse Load Match
	SCORR12	Reverse Transmission Tracking

EXAMPLE

```
CALC:DATA FDATA,Data(x)
calculate2:data sdata,data(r,i)
```

See another [example](#) using this command.

Return Type: [Block data](#)

Default - Not Applicable

CALCulate<cnum>:DATA:CUSTom <name>,<data> **Superseded**

Note: This command has been replaced by [CALC:DATA:](#) which can now be used with all PNA applications.

(Read-Write) Reads or writes data from a custom-named measurement buffer.

[See Critical Note](#)

Parameters

- <num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.
- <name> Name of the buffer to be read or written
- <data> Data to be read or written to the custom buffer. Format as one number per data point.

Examples

```
CALC:DATA:CUST 'VectorResult0',0,1,2,3,4,5 'Write  
CALC:DATA:CUST? 'VectorResult0' 'Read
```

Query Syntax CALCulate:DATA:CUSTom? <name>

Return Type Depends on [Form:Data](#)

Default Not Applicable

CALCulate<num>:DATA:CUSTom:CATalog? ~~Superseded~~

Note: This command has been replaced by [CALC:DATA:CAT](#) which can now be used with all PNA applications.

(Read-only) Reads the list of buffer names (comma separated list of string values) available from the selected parameter. Specify the measurement using [CALCulate:PARAMeter:SElect](#).

[See Critical Note](#)

Parameters

- <num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

Examples

```
CALC:DATA:CUST:CAT?  
calculate:data:custom:catalog?
```

Return Type String

Default Not Applicable

CALCulate<cnum>:DATA:SNP? <n> **Superseded**

Note: This command has been replaced by [CALC:DATA:SNP:PORTs?](#)

(Read-only) Reads SnP data from the selected measurement. [Learn more about SnP data.](#)

This command is valid **ONLY** with standard S-parameter measurements.

Notes

- This command returns SNP data without header information, and in columns, not in rows as .SnP files. This means that the data returned from this command sends all frequency data, then all Sx1 magnitude or real data, then all Sx1 phase or imaginary data, and so forth.
- To avoid frequency rounding errors, specify [FORM:DATA](#) <Real,64> or <ASCIi, 0>

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <n> Amount of data to return. If unspecified, <n> is set to 2. The number you specify must be less than or equal to the number of available ports on the PNA.

Choose from:

1 (S1P) returns 1-Port data for the active measurement if the active measurement is a reflection parameter such as S11 or S22. The behavior is UNDEFINED if the active measurement is a transmission parameter such as an S21.

2 (S2P) returns data for the four 2 port parameters associated with the current measurement. Default. Data that is not available is zero-filled.

3 (S3P) returns data for the nine 3 port parameters associated with the current measurement. Data that is not available is zero-filled.

4 (S4P) returns data for the sixteen 4 port parameters associated with the current measurement. Data that is not available is zero-filled.

SnP data can be output using several data formatting options. See [MMEM:STORe:TRACe:FORMat:SNP](#).

See also [MMEM:STOR <file>.<snp>](#)

Examples `CALC:PAR:DEF MyMeasurement, S11`

`CALC:PAR:SEL MyMeasurement`

`CALC:DATA:SNP? 1`

Return Type Depends on [FORMat:DATA](#).

Default Not Applicable

CALCulate<cnum>:DATA:SNP:PORTs? <"x,y,z">

Note: This command replaces [CALC:DATA:SNP?](#). This command is more explicit regarding the data to be returned, and works for PNAs with multiport test sets.

(Read-only) Reads SNP data from the selected measurement for the specified ports. [Learn more about SNP data.](#)

This command is valid **ONLY** with standard S-parameter measurements.

Notes

- This command returns SNP data without header information, and in columns, not in rows as .SnP files. This means that the data returned from this command sends all frequency data, then all Sx1 magnitude or real data, then all Sx1 phase or imaginary data, and so forth.
- To avoid frequency rounding errors, specify [FORM:DATA](#) <Real,64> or <ASCIi, 0>
- Data that is not available is zero-filled.
- For sweeps with a large number of data points, always follow this command with *OPC? [Learn more.](#)

[See Critical Note](#)

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<"x,y,z"> Comma or space delimited port numbers for which data is requested, enclosed in quotes.

SNP data can be output using several data formatting options. See [MMEM:STORe:TRACe:FORMat:SNP](#).

Examples `CALC:DATA:SNP:PORTs? "1,2,4,5,7" 'read data for these ports`

Return Type Depends on [FORMat:DATA](#)

Default Not Applicable

CALCulate<cnum>:DATA:SNP:PORTs:SAVE <"x,y,z">,<filename>

Note: This command replaces [MMEM:STOR sNp](#). This command is more explicit regarding the data to be saved, and works for PNAs with multiport test sets.

(Write-only) Saves SNP data from the selected measurement for the specified ports. [Learn more about SNP data.](#)

- The Normal vs Mixed Mode selection is NOT used as it is in the [Choose Ports dialog](#). Instead, data is returned as it is displayed on the trace. If the selected measurement is Mixed Mode (balanced), then balanced data is returned. If the selected measurement is an S-parameter, then S-parameter data is returned.
- This command is valid **ONLY** with the Standard measurement class (NOT applications).
- Data that is not available is zero-filled.
- For sweeps with a large number of data points, always follow this command with *OPC? [Learn more.](#)

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <"x,y,z"> **String** Comma or space delimited port numbers for which data is requested, enclosed in quotes.
- <filename> **String** Path, filename, and suffix of location to store the SNP data. The suffix is not checked for accuracy. If saving 2 ports, specify "filename.s2p"; If saving 4 ports, specify "filename.s4p.", and so forth.

SNP data can be output using several data formatting options. See [MMEM:STORe:TRACe:FORMat:SNP](#).

Examples

```
CALC:DATA:SNP:PORTs:Save '1,2,4','c:\users\public\network analyzer\documents\MyData.s3p';*OPC?
```

Return Type

Depends on [FORMat:DATA](#)

Default

Not Applicable

Last modified:

25-Nov-2013	Fixed path name for Win7
8-Aug-2013	Added Mixed / Normal note for SNP save
24-Sep-2012	Added notes for Calc:Data:SNP commands
11-Jul-2012	Calc:Data:SNP:Ports and SAVE- Added S-params only
18-May-2009	Superseded Custom commands
24-Mar-2009	Edited SNP note
26-Apr-2007	Added clarification to Calc:Data SDATA
9/18/06	MQ Added two SNP Ports commands for multiport

Calculate:Equation Commands

Controls Equation Editor capabilities.

CALCulate:EQUation:

LIBRARY

| [FUNCTIONS](#)

| [IMPORT?](#)

| [REMOVE](#)

STATE

TEXT

[VALID?](#)

Click on a [blue](#) keyword to view the command details.

see Also

- [Example Programs](#)
- [Learn about Equation Editor](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using [Calc:Par:MNUM](#) or [Calc:Par:Select](#). [Learn more](#).

CALCulate:EQUation:LIBRARY:FUNCTIONS <string>

(Read-only) Returns the functions in the specified DLL.

Parameters

<string> Full path and filename of the *.dll to be read.

Examples

```
functions = CALC:EQU:LIBR:FUNC "C:/Program Files/Agilent/Network Analyzer/UserFunctions/Expansion.dll"
```

Query Syntax CALCulate:EQUation:LIBRARY:FUNCTIONS?

Return Type Comma delimited string of function names.

Default Not Applicable

CALCulate:EQUation:LIBRARY:IMPORT <string>

(Read-Write) Imports the functions in the specified DLL and returns whether the functions have been imported into the PNA.

Parameters

<string> Full path and filename of the *.dll.

Examples

```
'Write - Imports functions
CALC:EQU:LIBR:IMPORT "C:/Program Files/Agilent/Network Analyzer/UserFunctions/Expansion.dll"

'Read if Imported

functions = CALC:EQU:LIBR:IMPORT "C:/Program Files/Agilent/Network Analyzer/UserFunctions/Expansion.dll"
```

Query Syntax CALCulate:EQUation:LIBRARY:IMPORT?

Returns the following:

1 - Imported

0 - NOT imported

Return Type Boolean

Default Not Applicable

CALCulate:EQUation:LIBRARY:REMOVe <string>

(Write-only) Removes an imported an Equation Editor DLL from the PNA.

Parameters

<string> Full path and filename of the *.dll.

Examples

```
CALC:EQU:LIBR:REM "C:/Program Files/Agilent/Network Analyzer/UserFunctions/Expansion.dll"
```

Query Syntax Not Applicable

Default Not Applicable

CALCulate<num>:EQUation[:STATE] <bool>

(Read-Write) Turns ON and OFF the equation on selected measurement for the specified channel. If the equation is not valid, then processing is not performed. Use [CALC:EQUation:VALid?](#) to ensure that the equation is valid.

[See Critical Note](#)

Parameters

<num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

<bool> **ON** (or 1) - turns equation ON.

OFF (or 0) - turns equation OFF.

Examples

```
CALC:EQU 1  
calculate2:equation:state 0
```

Query Syntax CALCulate<num>:EQUation[:STATE]?

Return Type Boolean

Default OFF (0)

CALCulate<num>:EQUation:TEXT <string>

(Read-Write) Specifies an equation or expression to be used on the selected measurement for the specified channel.

[See Critical Note](#)

Parameters

<num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

<string> Any valid equation or expression. [See Equation Editor](#).

Examples

```
'Equation (includes '=')
CALC:EQU:TEXT "foo=S11/S21"

'Expression
calculate2:equation:text "S11/S21"
```

Query Syntax CALCulate<num>:EQUation:TEXT?

Return Type String

Default Not Applicable

CALCulate<num>:EQUation:VALid?

(Read-Only) Returns a boolean value to indicate if the current equation on the selected measurement for the specified channel is valid. For equation processing to occur, the equation must be valid and ON ([CALC:EQU:STAT 1](#)).

[See Critical Note](#)

Parameters

<num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

Examples

```
CALC:EQU:VAL?
calculate2:equation:valid?
```

Return Type Boolean

- 1 - equation is valid
- 0 - equation is NOT valid

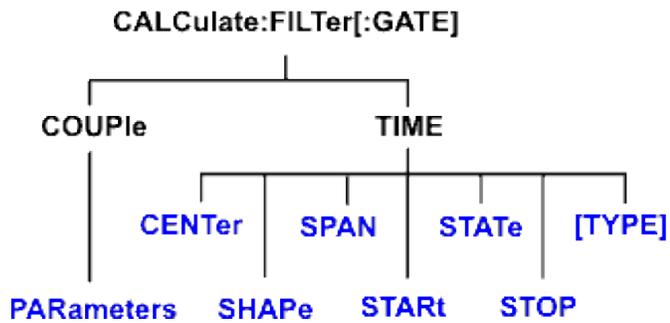
Default Not Applicable

Last Modified:

30-Jan-2014	Several edits to cnums
10-Jan-2011	New topic

Calculate:Filter Commands

Controls the gating function used in time domain measurements. The gated range is specified with either (start / stop) or (center / span) commands.



Click on a [blue](#) keyword to view the command details.

see Also

- [Example Programs](#)
- [Learn about Gating](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using [Calc:Par:MNUM](#) or [Calc:Par:Select](#). [Learn more](#).

CALCulate<num>:FILTerf:GATEf:COUPlE:PARAmeters <num>

(Read-Write) Specifies the time domain gating parameters to be coupled. The settings for those parameters will be copied from the selected measurement to all other measurements on the channel.

- To enable Trace Coupling, use [SENS:COUP:PAR](#)
- To specify Transform parameters to couple, use [CALC:TRAN:COUP:PAR](#)

Learn more about [Time Domain Trace Coupling](#)

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <num> (Numeric) Parameters to couple. To specify more than one parameter, add the numbers.

1 - Gating Stimulus (Start, Stop, Center, and Span TIME settings.)

2 - Gating State (ON / OFF)

4 - Gating Shape (Minimum, Normal, Wide, and Maximum)

8 - Gating Type (Bandpass and Notch)

Examples

```
'To couple all parameters:
```

```
CALC:FILT:COUP:PAR 15
```

```
'To couple Stimulus and Type:
```

```
calculate2:filter:gate:couple:parameters 9
```

Query Syntax CALCulate<cnum>:FILTer:GATE:COUPle:PARAmeters?

Return Type Numeric

Default 13 (All parameters except 2 - Gating State)

CALCulate<cnum>:FILTer[:GATE]:TIME:CENTer <num>

(Read-Write) Sets the gate filter center time.

[See Critical Note](#)

Parameters

- <num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.
- <num> Center time in seconds; Choose any number between:
 $\pm (\text{number of points}-1) / \text{frequency span}$

Note: This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
CALC:FILT:GATE:TIME:CENT -5 ns  
calculate2:filter:time:center maximum
```

Query Syntax CALCulate<num>:FILTer[:GATE]:TIME:CENTer?

Return Type Numeric

Default 0

CALCulate<num>:FILTer[:GATE]:TIME:SHAPE <char>

(Read-Write) Sets the gating filter shape when in time domain.

[See Critical Note](#)

Parameters

- <num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.
- <char> Choose from
MAXimum - the widest gate filter available
WIDE -
NORMAl -
MINimum - the narrowest gate filter available

Examples

```
CALC:FILT:GATE:TIME:SHAP MAX  
calculate2:filter:time:shape normal
```

Query Syntax CALCulate<num>:FILTer[:GATE]:TIME:SHAPE?

Return Type Character

Default NORMAl

CALCulate<cnum>:FILTer[:GATE]:TIME:SPAN <num>

(Read-Write) Sets the gate filter span time.

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <num> Time span in seconds; Choose any number between: 0 and $2 * [(number\ of\ points - 1) / frequency\ span]$

Note: This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
CALC:FILT:GATE:TIME:SPAN 5 ns
calculate2:filter:time:span maximum
```

Query Syntax CALCulate<cnum>:FILTer[:GATE]:TIME:SPAN?

Return Type Numeric

Default 20 ns

CALCulate<cnum>:FILTer[:GATE]:TIME:STATe <boolean>

(Read-Write) Turns gating state ON or OFF.

[See Critical Note](#)

Note: Sweep type must be set to LInear Frequency in order to use Transform Gating.

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <boolean> **ON** (or 1) - turns gating ON.
OFF (or 0) - turns gating OFF.

Examples

```
CALC:FILT:TIME:STAT ON
calculate2:filter:gate:time:state off
```

Query Syntax CALCulate<cnum>:FILTer[:GATE]:TIME:STATe?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

CALCulate<cnum>:FILTer[:GATE]:TIME:STARt <num>

(Read-Write) Sets the gate filter start time.

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <num> Start time in seconds; any number between:
 $\pm (\text{number of points}-1) / \text{frequency span}$

Note: This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
CALC:FILT:TIME:STAR 1e-8  
calculate2:filter:gate:time:start minimum
```

Query Syntax CALCulate<cnum>:FILTer[:GATE]:TIME:STARt?

Return Type Numeric

Default 10 ns

CALCulate<cnum>:FILTer[:GATE]:TIME:STOP <num>

(Read-Write) Sets the gate filter stop time.

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <num> Stop time in seconds; any number between:
 $\pm (\text{number of points}-1) / \text{frequency span}$

Note: This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
CALC:FILT:TIME:STOP -1 ns  
calculate2:filter:gate:time:stop maximum
```

Query Syntax CALCulate<cnum>:FILTer[:GATE]:TIME:STOP?

Return Type Numeric

Default 10 ns

CALCulate<num>:FILTer[:GATE]:TIME[:TYPE] <char>

(Read-Write) Sets the type of gate filter used.

[See Critical Note](#)

Parameters

<num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

<char> Choose from:

BPASs - Includes (passes) the range between the start and stop times.

NOTCh - Excludes (attenuates) the range between the start and stop times.

Examples

```
CALC:FILT:TIME BPAS  
calculate2:filter:gate:time:type notch
```

Query Syntax CALCulate<num>:FILTer[:GATE]:TIME[:TYPE]?

Return Type Character

Default BPAS

Calculate:Format Commands

CALCulate:

FORMat

UNIT

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using [Calc:Par:MNUM](#) or [Calc:Par:Select](#). [Learn more](#).

See Also

- [Example](#) using this command.
 - [Learn About Data Format](#)
 - [Synchronizing the PNA and Controller](#)
-

CALCulate<cnun>:FORMat <char>

(Read-Write) Sets the display format for the measurement.

[See Critical Note](#)

Parameters

<cnun> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnun> is set to 1.

<char> Choose from:

- MLINear
- MLOGarithmic
- PHASe
- UPHase 'Unwrapped phase
- IMAGinary
- REAL
- POLar

- SMITh
- SADMittance 'Smith Admittance
- SWR
- GDELay 'Group Delay
- KELVin
- FAHRenheit
- CELSius

Examples `CALC:FORM MLIN`
`calculate2:format polar`

Query Syntax `CALCulate<cnum>:FORMat?`

Return Type Character

Default MLINear

CALCulate<cnum>:FORMat:UNIT <dataFormat>, <units>

(Read-Write) Sets and returns the units for the specified data format. Measurements with display formats other than those specified are not affected.

[See Critical Note](#)

Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<dataFormat>` Choose MLOG.
- `<units>` For unratioed measurements, choose from:
 - **DBM** Units are displayed in dBm. 0 dBm = 0.001 watt
 - **DBMV** Units are displayed in dBmV. 0 dBmV = 0.001 volt

Examples `CALC:FORM MLOG, DBM`
`calculate2:format mlog, dbmv`

Query Syntax `CALCulate<cnum>:FORMat:UNIT? <dataFormat>`

Return Type Character

Default DBM

Last Modified:

25-Aug-2008	Added Units
1-Oct-2007	Added temperature formats

Calculate:FSimulator Commands

Specifies settings and fixturing for Balanced Measurements.

CALCulate:FSIMulator

[BALun](#) [More commands](#)

[EMBed](#) [More commands](#)

[SENDEd](#) [More commands](#)

[SNP:EXTRapolate](#)

[STATe](#)

Click a [blue](#) keyword to view the command details.

See Also

- [Example Programs](#)
 - [SCPI Command Tree](#)
-

CALCulate<cnum>:FSIMulator:SNP:EXTRapolate <bool>

(Read-Write) Turns ON and OFF SNP file extrapolation for both 2-port and 4-port embedding/de-embedding. [Learn more.](#)

Note: This command affects ALL measurements on the specified channel.

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<bool> Choose from:

ON or 1 - Turns Extrapolation ON

OFF or 0 - Turns Extrapolation OFF

Examples

```
CALC:FSIM:SNP:EXTR 1  
calculate2:fsimulator:snp:extrapolate 0
```

Query Syntax CALCulate<cnum>:FSIMulator:SNP:EXTRapolate?

Return Type Boolean

Default OFF

CALCulate<cnum>:FSIMulator:STATe <bool>

(Read-Write) Turns all three fixturing functions (de-embedding, port matching, impedance conversion) ON or OFF for all ports on the specified channel. Does not affect port extensions.

Note: This command affects ALL measurements on the specified channel.

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<bool> Choose from:

ON or 1 - Turns Fixturing ON

OFF or 0 - Turns Fixturing OFF

Examples

```
CALC:FSIM:STAT 1  
calculate2:fsimulator:state 0
```

Query Syntax CALCulate<cnum>:FSIMulator:STATe?

Return Type Boolean

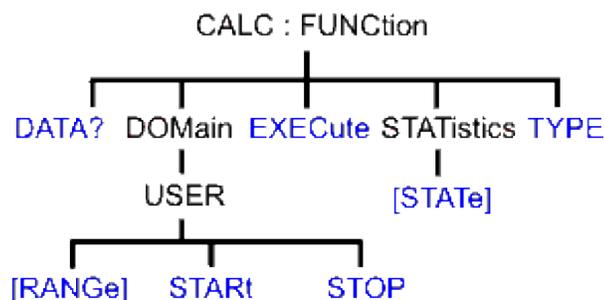
Default OFF

Last Modified:

4-Nov-2013 Fixed link

16-Nov-2010 Added extrapolate

Calculate:Function Commands



Click on a [blue](#) keyword to view the command details.

see Also

- [Example Programs](#)
- [Learn about Trace Statistics](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using [Calc:Par:MNUM](#) or [Calc:Par>Select](#). [Learn more](#).

CALCulate<num>:FUNCtion:DATA?

(Read-only) Returns the trace statistic data for the selected statistic type for the specified channel. Select the type of statistic with [CALC:FUNC:TYPE](#).

[See Critical Note](#)

Parameters

<num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

Return Type Depends on [FORM:DATA](#)

Example `CALCulate2:FUNCtion:DATA?`

Default Not applicable

CALCulate<num>:FUNCtion:DOMain:USER[:RANGe] <range>

(Read-Write) Sets the range used to calculate trace statistics. Each channel has 16 user ranges. The x-axis range is specified with the [CALC:FUNC:DOM:USER:START](#) and [STOP](#) commands.

[See Critical Note](#)

Parameters

<num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

<range> Range number. Choose from: **0** to **16**

0 is Full Span of the current x-axis range

1 to 16 are user-specified ranges

Examples

```
CALC:FUNC:DOM:USER 4  
calculate2:function:domain:user:range 0
```

Query Syntax CALCulate<num>:FUNCTION:DOMAIN:USER[:RANGE]?

Return Type Numeric

Default 0 - Full Span

CALCulate<num>:FUNCTION:DOMAIN:USER:START <range>, <start>

(Read-Write) Sets the start of the specified user-domain range.

To apply this range, use [CALC:FUNC:DOM:USER](#)

To set the stop of the range, use [CALC:FUNC:DOM:USER:STOP](#).

[See Critical Note](#)

Note: This command does the same as [CALC:MARK:FUNC:DOM:USER:STAR](#)

Parameters

<num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

<range> Range number that will receive the start value. Choose an integer between **1** and **16**

<start> Start value of the specified range. Choose a real number between: the analyzer's **Minimum** and **Maximum** x-axis value.

Examples

```
CALC:FUNC:DOM:USER:STAR 1,1e9  
calculate2:function:domain:user:start 2,2e9
```

Query Syntax CALCulate<cnum>:FUNCTION:DOMAIN:USER:START? <range>

Return Type Numeric

Default The analyzer's **Minimum** x-axis value

CALCulate<cnum>:FUNCTION:DOMAIN:USER:STOP <range>, <stop>

(Read-Write) Sets the stop value of the specified user-domain range.

To apply this range, use [CALC:FUNC:DOM:USER](#).

To set the start of the range, use [CALC:FUNC:DOM:USER:START](#)

[See Critical Note](#)

Note: This command does the same as [CALC:MARK:FUNC:DOM:USER:STOP](#)

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<range> Range number that will receive the stop value. Choose an integer between **1** and **16**

<stop> Stop value of the specified range. Choose a real number between: the analyzer's **Minimum** and **Maximum** x-axis value.

Examples

```
CALC:FUNC:DOM:USER:STOP 4,5e9  
calculate2:function:domain:user:stop 3,8e9
```

Query Syntax CALCulate<cnum>:FUNCTION:DOMAIN:USER:STOP? <range>

Return Type Numeric

Default The analyzer's **Maximum** x-axis value

CALCulate<cnum>:FUNCTION:EXECute

(Write-only) For the active trace of specified channel, executes the statistical analysis specified by the [CALC:FUNC:TYPE](#) command.

[See Critical Note](#)

Parameters

<num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

Examples

```
CALC:FUNC:EXEC  
calculate2:function:execute
```

Query Syntax Not Applicable

Default Not Applicable

CALCulate<num>:FUNCTION:STATistics[:STATE] <ON|OFF>

(Read-Write) Displays and hides the trace statistics (peak-to-peak, mean, standard deviation) on the screen.

The analyzer will display either measurement statistics or Filter Bandwidth statistics; not both.

[See Critical Note](#)

Parameters

<num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

<ON|OFF> ON - Displays trace statistics

OFF - Hides trace statistics

Examples

```
CALC:FUNC:STAT ON  
calculate2:function:statistics:state off
```

Query Syntax CALCulate<num>:FUNCTION:STATistics[:STATE]?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF (0)

CALCulate<num>:FUNCTION:TYPE <char>

(Read-Write) Sets statistic TYPE that you can then query using [CALC:FUNCtion:DATA?](#).

Note: In PNA releases 4.2 and prior, this command applied the statistic type to all measurements. Now, this command affects only the selected measurement on the specified channel.

[See Critical Note](#)

Parameters

<num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

<char> Choose from:

PTPeak - the difference between the max and min data points on the trace.

STDEV - standard deviation of all data points on the trace

MEAN - mean (average) of all data points on the trace

MIN - lowest data point on the trace

MAX - highest data point on the trace

Examples

```
CALC:FUNC:TYPE PTP  
calculate2:function:type stdev
```

Query Syntax CALCulate<num>:FUNCtion:TYPE?

Return Type Character

Default PTPeak

Calc:GCData Commands

Reads Gain Compression data from the current Gain Compression acquisition.

CALCulate:GCData:

[DATA](#)

[IMAG](#)

[ITERations](#)

[REAL](#)

Click on a [blue](#) keyword to view the command details.

Other Gain Compression commands

The calibration commands listed in this topic are supplemental to the Guided Cal commands.

- [CALC:CUSTom:DEFine](#) - creates a gain compression measurement.
- [SENS:GCSetup](#) - Most Gain Compression settings.
- [CALC:GCMeas:ANAL](#) - Gain Compression Analysis settings
- Gain compression data can also be saved to a *.csv file. [Learn how.](#)

See Also

- [Example Programs](#)
 - [Learn about Gain Compression Application](#)
 - [Synchronizing the PNA and Controller](#)
 - [SCPI Command Tree](#)
-

CALCulate<ch>:GCData:DATA? <param>

(Read-Only) Returns measurement data at all frequency and power data points for GCA SMART sweeps and 2D sweeps.

- When using SMART sweep, ALL data is returned including ALL background iteration sweeps. Use [CALC:GCD:ITER](#) to determine the number of iteration sweeps. The number of data points that are returned is always going to be number of frequency points times the number of iteration sweeps.
- When using 2D sweeps, ALL data is returned. The number of data points returned / freq may vary. [Learn](#)

[more.](#)

Use [Calc:Data?](#) to return just the displayed data results (not the background sweeps).

A compression parameter must be present. [Learn more.](#)

The format of the data is the same as the format of the measurement that you select using [Calc:Par:Select](#). If the measurement is scalar, than one number is returned per sweep per data point. If complex (such as Smith Chart format) than both real and imaginary numbers are returned.

If correction is on, corrected data are returned. Otherwise, raw data are returned.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <param> (String) Parameter to read. NOT Case-sensitive. The specified parameter need NOT be displayed or selected. However, a compression parameter must be present. [Learn more.](#)

Choose from:

- **"pin"** - (CompIn21) Input power at the compression point.
- **"pout"** - (CompOut21) Output power at the compression point.
- **"gain"** - (CompGain21) Device gain (S21) at the compression point.
- **"inputmatch"** - (CompS11) Input match at the compression point.
- **"DeltaGain"** - (DeltaGain21) Measured Gain (watts) / Ref Gain (watts). [Learn more.](#)
- **"AI1"** and **"AI2"** - ADC measurements at the specified compression level. [Learn more.](#)

[Learn more about GCA parameters.](#)

Examples

```
data = CALC:GCD:DATA? "pin"  
data = calculate:gcddata:data? "pout"
```

Return Type Array of data

Default Not Applicable

CALCulate<ch>:GCDData:IMAG? <char>, <dpoint>, <param>

(Read-Only) For a specified data point, returns the imaginary part of the specified Gain Compression data. If correction is on, corrected data are returned. Otherwise, raw data are returned. Can be used

with Smart and 2D sweeps.

- For SMART sweep, the number of data points that are returned is always going to be the number of iteration sweeps. Use [CALC:GCD:ITER](#) to determine the number of iteration sweeps.
- For 2D sweeps, the number of data points returned / freq may vary. [Learn more.](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <char> Choose from:
- **FREQuency** - for the specified frequency data point, returns all of the measured data for each power stimulus.
 - **POWer** - for the specified power data point, returns all of the measured data for each frequency stimulus.
- <dPoint> Data point (FREQ or POWer) for which data is returned.
- <param> (String) Parameter to read. NOT Case-sensitive. The specified parameter need NOT be displayed. However, a compression parameter must be present. [Learn more.](#)
- **"pin"** - (CompIn21) Input power at the compression point.
 - **"pout"** - (CompOut21) Output power at the compression point.
 - **"gain"** - (CompGain21) Device gain (S21) at the compression point.
 - **"inputmatch"** - (CompS11) Input match at the compression point.
 - **"DeltaGain"** - (DeltaGain21) Measured Gain (watts) / Ref Gain (watts). [Learn more.](#)
 - **"AI1"** and **"AI2"** - ADC measurements at the specified compression level. [Learn more.](#)

Examples

For the fifth frequency data point, returns 'Power Output' imaginary (phase) data from all power stimulus values.

For SmartSweep, if there are 30 power sweep points, 30 values are returned.

For 2D sweeps, 30 or 31 power sweep points may be returned. [Learn more.](#)

```
data = CALC:GCD:IMAG? FREQ,5,"pout"
```

Return Type Array of data

Default Not Applicable

CALCulate<num>:GCDData:ITERations?

(Read-only) In a SMART sweep, returns the max number of iterations that it took for ALL frequencies to converge. Use this number to determine the size of the block data that is returned from Gain Compression SMART sweep data queries.

For a 2D sweep, returns the number of power points.

Parameters

<num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

Examples

```
data = CALC:GCD:ITER?
```

Return Type Numeric

Default Not Applicable

CALCulate<ch>:GCDData:REAL? <char>, <dpoint>, <param>

(Read-Only) For a specified data point, returns the real part of the Gain Compression data. If correction is on, corrected data are returned. Otherwise, raw data are returned. Can be used with Smart and 2D sweeps.

- For SMART sweep, the number of data points that are returned is always going to be the number of iteration sweeps. Use [CALC:GCD:ITER](#) to determine the number of iteration sweeps.
- For 2D sweeps, the number of data points returned / freq may vary. [Learn more.](#)

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<char> Choose from:

- **FREQuency** - for the specified frequency data point, returns all of the measured data for each power stimulus.
- **POWer** - for the specified power data point, returns all of the measured data for each frequency stimulus.

<dPoint> Data point (FREQu or POWer) for which data is returned.

<param> (String) Parameter to read. NOT Case-sensitive. The specified parameter need NOT be displayed. However, a compression parameter must be present. [Learn more.](#)

- **"pin"** - (CompIn21) Input power at the compression point.
- **"pout"** - (CompOut21) Output power at the compression point.
- **"gain"** - (CompGain21) Device gain (S21) at the compression point.
- **"inputmatch"** - (CompS11) Input match at the compression point.
- **"DeltaGain"** - (DeltaGain21) Measured Gain (watts) / Ref Gain (watts). [Learn more.](#)
- **"AI1"** and **"AI2"** - ADC measurements at the specified compression level. [Learn more.](#)

Examples

For the fifth frequency data point, returns 'Power Output' real data from all power stimulus values.

For SmartSweep, if there are 30 power sweep points, 30 values are returned.

For 2D sweeps, 30 or 31 power sweep points may be returned. [Learn more.](#)

```
data = CALC:GCD:REAL? FREQ,5,"pout"
```

Return Type Array of data

Default Not Applicable

Last Modified:

7-May-2012	Clarify Iterations queries. (DS)
30-Aug-2011	Edited for 2D sweep variation
13-May-2011	Many clarification edits
30-Mar-2009	Added ADC and DeltaGain params
24-Apr-2008	Edited for any GCA sweep
18-Oct-2007	MX New topic

Gain Compression Analysis Commands

Sets and reads Gain Compression Analysis controls.

CALCulate:GCMeas:ANALysis

[CWFRequency](#)

[ENABle](#)

[ISDisfreq](#)

[XAXis](#)

Click on a [blue](#) keyword to view the command details.

Other Gain Compression commands

The calibration commands listed in this topic are supplemental to the Guided Cal commands.

- [CALC:CUSTom:DEFine](#) - creates a gain compression measurement.
- [SENS:GCSetup](#) - Most Gain Compression settings.
- [GC:DATA](#) - Gain Compression data commands
- Gain compression data can also be saved to a *.csv file. [Learn how.](#)

See Also

- [Example Programs](#)
- [Learn about Compression Analysis](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

CALCulate<cnum>:GCMeas:ANALysis:ENABle <bool>

(Read-Write) Enables and disables a compression analysis trace.

Parameters

<cnun> Channel number of the GCA measurement. There must be a selected measurement on that channel using [Calc:Par:Sel](#). If unspecified, <cnun> is set to 1.

<bool> **ON** (or 1) - Enable compression analysis.

OFF (or 0) - Disable compression analysis.

Examples

```
CALC:GCM:ANAL:ENAB ON  
calculate2:gcmeas:analysis:enable off
```

Query Syntax CALCulate<cnun>:GCMeas:ANALysis:ENABLE?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

CALCulate<cnun>:GCMeas:ANALysis:CWFRequency <num>

(Read-Write) Set and return the CW frequency for a compression analysis trace.

Parameters

<cnun> Channel number of the GCA measurement. There must be a selected measurement on that channel using [Calc:Par:Sel](#). If unspecified, <cnun> is set to 1.

<num> CW frequency in Hz. Choose a frequency within the range of the gain compression channel.

Examples

```
CALC:GCM:ANAL:CWFR 1e9  
calculate2:gcmeas:analysis:cwfrequency 1e10
```

Query Syntax CALCulate<cnun>:GCMeas:ANALysis:CWFRequency?

Return Type Numeric

Default Not Applicable

CALCulate<cnun>:GCMeas:ANALysis:ISDisfrequency <bool>

(Read-Write) Sets and returns whether the CW frequency for the compression analysis trace can be set to only the discrete frequencies or provides interpolation.

Parameters

- <num> Channel number of the GCA measurement. There must be a selected measurement on that channel using [Calc:Par:Sel](#). If unspecified, <num> is set to 1.
- <bool> **ON** (or 1) - Discrete data points only.
OFF (or 0) - Interpolated data points.

Examples

```
CALC:GCM:ANAL:ISD ON  
calculate2:gcmeas:analysis:isdisfrequency off
```

Query Syntax CALCulate<num>:GCMeas:ANALysis:ISDisfrequency?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

CALCulate<num>:GCMeas:ANALysis:XAXis <char>

(Read-Write) Sets and returns the type of data to display on the x-axis of a compression analysis trace.

Parameters

- <num> Channel number of the GCA measurement. There must be a selected measurement on that channel using [Calc:Par:Sel](#). If unspecified, <num> is set to 1.
- <bool> Data to display on X-axis. Choose from:
- **PIN** - Input power to the DUT.
 - **PSource** - power from the source.

Examples

```
CALC:GCM:ANAL:XAX PIN  
calculate2:gcmeas:analysis:xaxis psource
```

Query Syntax CALCulate<num>:GCMeas:ANALysis:XAXis?

Return Type Character

Default PIN

13-Aug-2009 MX New topic

Group Delay Aperture Commands

Controls the Aperture setting used to make Group Delay measurements.

CALCulate:GDElay

[FREQuency](#)

[PERCent](#)

[POINts](#)

Click on a [blue](#) keyword to view the command details.

see Also

- [Learn about Group Delay Aperture](#)
- [Example Programs](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using [Calc:Par:MNUM](#) or [Calc:Par>Select](#). [Learn more](#).

CALCulate<cnum>:GDElay:FREQuency <value>

(Read-Write) Sets group delay aperture using a fixed frequency range.

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <value> Frequency range (in Hz) to use for the aperture setting. Choose between the equivalent of two data points and the channel frequency span.

Examples

```
CALC:GDEL:FREQ 1E6
```

Query Syntax CALCulate<cnum>:GDElay:FREQuency?

Return Type Numeric

Default Frequency range that equates to 11 points. This can be changed to two points with a [preference setting](#).

CALCulate<cnum>:GDElay:PERCent <value>

(Read-Write) Sets group delay aperture using a percent of the channel frequency span.

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <value> Percent of frequency span to use for the aperture setting. Choose between the equivalent of two data points and 100 percent of the channel frequency span.

Examples

```
'set to 25 percent of the channel frequency span
```

```
CALC:GDEL:PERC 25
```

Query Syntax CALCulate<cnum>:GDElay:PERCent?

Return Type Numeric

Default Percent of frequency span that equates to 11 points. This can be changed to two points with a [preference setting](#).

CALCulate<cnum>:GDElay:POINts <value>

(Read-Write) Sets group delay aperture using a fixed number of data points.

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <value> Number of data points to use for the aperture setting. Choose between two points and the number of points in the channel.

Examples

```
'set to 25 data points
```

```
CALC:GDEL:POIN 25
```

Query Syntax CALCulate<cnum>:GDElay:POINts?

Return Type Numeric

Default 11 points. This can be changed to two points with a [preference setting](#).

Last Modified:

23-Feb-2010 MX New topic

Calc:Limit Commands

Controls the limit segments used for pass / fail testing.

CALCulate:LIMit:

DATA

| [DELeTe](#)

DISPlay

| [\[STATe\]](#)

FAIL?

SEGMENT

| [AMPLitude](#)

| [STARt](#)

| [STOP](#)

| [STIMulus](#)

| [STARt](#)

| [STOP](#)

| [TYPE](#)

SOUND

| [\[STATe\]](#)

[\[STATe\]](#)

Click on a [blue](#) keyword to view the command details.

see Also

- [Example Programs](#)
- [Learn about Limit Lines](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

Critical Note: CALCulate commands act on the selected measurement. You can select one

measurement for each channel using [Calc:Par:MNUM](#) or [Calc:Par>Select](#). [Learn more](#).

CALCulate<cnum>:LIMit:DATA <block>

(Read-Write) Sets data for limit segments.

[See Critical Note](#)

Parameters

<cnum> Channel number of the measurement for which limit lines are to be set. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<block> Data for all limit segments in REAL,64 format. The following is the data format for 1 segment:

Type,BegStim, EndStim, BegResp,EndResp

Type Type of limit segment. Choose from
0 - Off
1 - Max
2 - Min

BegStim Start of X-axis value (freq, power, time)

EndStim End of X-axis value

BegResp Y-axis value that corresponds with Start of X-axis value

EndResp Y-axis value that corresponds with End of X-axis value

Examples

The following writes three max limit segments for a bandpass filter.

```
CALC:LIM:DATA 1,3e5,4e9,-60,0,1,4e9,7.5e9,0,0,1,7.5e9,9e9,0,-30
```

Query Syntax CALCulate<cnum>:LIMit:DATA?

Return Type Depends on [FORM:DATA](#) - All 100 predefined limit segments are returned.

Default 100 limit segments - all values set to 0

CALCulate<cnum>:LIMit:DATA:DELeTe

(Write-only) Deletes all limit line data for the selected measurement on the specified channel.

[See Critical Note](#)

Parameters

<num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

Examples `CALC2:LIM:DATA:DEL`

Query Syntax Not Applicable

Default Not Applicable

CALCulate<num>:LIMit:DISPlay[:STATe] <ON | OFF>

(Read-Write) Turns the display of limit segments ON or OFF (if the data trace is turned ON).

[See Critical Note](#)

Parameters

<num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

<ON | OFF> **ON** (or 1) - turns the display of limit segments ON.
OFF (or 0) - turns the display of limit segments OFF.

Examples `CALC:LIM:DISP:STAT ON`
`calculate2:limit:display:state off`

Query Syntax CALCulate<num>:LIMit:DISPlay[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

CALCulate<num>:LIMit:FAIL?

(Read-only) Returns the Pass / Fail status of the limit line test. Returns 1 (Fail) if any data point fails for any limit segment.

Limit display (CALC:LIM:DISP) does NOT have to be ON.

[See Critical Note](#)

Parameters

<num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

Examples

```
CALC:LIM:FAIL?
```

Return Type Boolean

- **0** is returned when **Pass**
- **1** is returned when **Fail**

Default Not Applicable

CALCulate<num>:LIMit:SEGment<num>:AMPLitude:STARt <num>

(Read-Write) Sets the start (beginning) of the Y-axis amplitude (response) value.

[See Critical Note](#)

Parameters

<num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

<num> Segment number; if unspecified, value is set to 1.

<num> Choose any number between: **-500** and **500**

Display value is limited to the Maximum and Minimum displayed Y-axis values.

Examples

```
CALC:LIM:SEG1:AMPL:STAR 10  
calculate2:limit:segment2:amplitude:start 10
```

Query Syntax CALCulate<num>:LIMit:SEGment<num>AMPLitude:STARt?

Return Type Numeric

Default 0

CALCulate<cnum>:LIMit:SEGMENT<snum>:AMPLitude:STOP <num>

(Read-Write) Sets the stop (end) of the Y-axis amplitude (response) value.

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <snum> Segment number; if unspecified, value is set to 1.
- <num> Choose any number between: **-500** and **500**

Display value is limited to the Maximum and Minimum displayed Y-axis values.

Examples

```
CALC:LIM:SEGM1:AMPL:STOP 10  
calculate2:limit:segment2:amplitude:stop 10
```

Query Syntax CALCulate<cnum>:LIMit:SEGMENT<snum>AMPLitude:STOP?

Return Type Numeric

Default 0

CALCulate<cnum>:LIMit:SEGMENT<snum>:STIMulus:STARt <num>

(Read-Write) Sets the start (beginning) of the X-axis stimulus value.

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <snum> Segment number; if unspecified, value is set to 1.
- <num> Choose any number within the X-axis span of the analyzer.

Examples

```
CALC:LIM:SEGM1:STIM:STAR 10  
calculate2:limit:segment2:stimulus:start 10
```

Query Syntax CALCulate<cnum>:LIMit:SEGMENT<snum>STIMulus:STARt?

Return Type Numeric

Default 0

CALCulate<cnum>:LIMit:SEGMENT<snum>:STIMulus:STOP <num>

(Read-Write) Sets the stop (end) of the X-axis stimulus value.

[See Critical Note](#)

Parameters

- <num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.
- <snum> Segment number; if unspecified, value is set to 1.
- <num> Choose any number within the X-axis span of the analyzer.

Examples

```
CALC:LIM:SEGM1:AMPL:STOP 10  
calculate2:limit:segment2:stimulus:stop 10
```

Query Syntax CALCulate<num>:LIMit:SEGMent<snum>STIMulus:STOP?

Return Type Numeric

Default 0

CALCulate<num>:LIMit:SEGMent<snum>:TYPE <char>

(Read-Write) Sets the type of limit segment.

[See Critical Note](#)

Parameters

- <num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.
- <snum> Segment number. Choose any number between:
1 and **100**
If unspecified, value is set to 1.
- <char> Choose from:
LMAX - a MAX limit segment. Any response data exceeding the MAX value will fail.
LMIN - a MIN limit segment. Any response data below the MIN value will fail.
OFF - the limit segment (display and testing) is turned OFF.

Examples

```
CALC:LIM:SEGM:TYPE LMIN  
calculate2:limit:segment3:type lmax
```

Query Syntax CALCulate<num>:LIMit:SEGMent<snum>:TYPE?

Return Type Character

Default OFF

CALCulate<cnum>:LIMit:SOUNd[:STATe] <ON | OFF>

(Read-Write) Turns limit testing fail sound ON or OFF.

[See Critical Note](#)

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<ON | OFF> **ON** (or 1) - turns sound ON.
OFF (or 0) - turns sound OFF.

Examples

```
CALC:LIM:SOUN ON  
calculate2:limit:sound:state off
```

Query Syntax CALCulate<cnum>:LIMit:SOUNd[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

CALCulate<cnum>:LIMit[:STATe] <ON | OFF>

(Read-Write) Turns limit segment **testing** ON or OFF.

- Use [CALC:LIM:DISP](#) to turn ON and OFF the **display** of limit segments.
- If using [Global Pass/Fail](#) status, trigger the PNA AFTER turning Limit testing ON.

[See Critical Note](#)

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<ON | OFF> **ON** (or 1) - turns limit testing ON.
OFF (or 0) - turns limit testing OFF.

Examples

```
CALC:LIM:STAT ON  
calculate2:limit:state off
```

Query Syntax CALCulate<cnum>:LIMit:STATe?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

Last Modified:

10-Jan-2011 Minor edits

31-Oct-2008 Added Delete and Fail commands (8.33)

Calculate:Marker Commands

Controls the marker settings used to remotely output specific data to the computer.

CALCulate:MARKer:

AOFF

BUCKet

BWIDth

COMPression

| **LEVel**

| **PIN?**

| **POUT?**

COUPling

| **[STATe]**

DELTA

DISCcrete

DISTance

FORMat

FUNCTION

| **APEak**

| **EXCursion**

| **THReshold**

| **DOMain**

| **USER**

| **STARt**

| **STOP**

| **EXECute**

| **[SElect]**

| **TRACking**

[PNOP more commands](#)

[PSATuration more commands](#)

REFerence

[|\[STATe\]](#)

[|X](#)

[|Y?](#)

SET

[\[STATe\]](#)

TARGet

TYPE

X

Y?

Click on a [blue](#) keyword to view the command details.

See Also

- [Marker example program](#)
- Marker Readout [number](#) and [size](#) commands.
- [Learn about Markers](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using [Calc:Par:MNUM](#) or [Calc:Par>Select](#). [Learn more](#).

Note: The Reference Marker is Marker Number 10

CALCulate<cnum>:MARKer:AOff

(Write-only) Turns all markers off for selected measurement.

[See Critical Note](#)

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

Examples

```
CALC:MARK:AOFF  
calculate2:marker:aoff
```

Query Syntax Not applicable

Default Not applicable

CALCulate<cnum>:MARKer<n>:BUCKet <num>

(Read-Write) Sets and reads the data point (bucket) number of the trace on which the marker resides. When the markers are [interpolated \(non-discrete\)](#), the returned value is the nearest marker bucket position.

[See Critical Note](#)

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<n> Marker number to move or query. The marker must already exist. If unspecified, <n> is set to 1.

<num> Data point (bucket) number. Choose any data point between: 0 and the number of data points minus 1.

Examples

```
CALC:MARK:BUCK 5  
calculate2:marker2:bucket 200
```

Query Syntax CALCulate<cnum>:MARKer<n>:BUCKet?

Return Type Integer

Default The first marker is set to the middle of the span. Subsequent markers are set to the bucket number of the previously active marker.

CALCulate<cnum>:MARKer:BWIDth <num>

(Read-Write) Turns on and sets markers 1 through 4 to calculate filter bandwidth. The <num> parameter sets the value below the maximum bandwidth peak that establishes the bandwidth of a filter. For example, if you want to determine the filter bandwidth 3 db below the bandpass peak value, set <num> to -3.

To turn off the Bandwidth markers, either turn them off individually or turn them [All Off](#).

The analyzer screen will show either Bandwidth statistics OR Trace statistics; not both.

To search a User Range with the bandwidth search, first activate marker 1 and set the desired [User Range](#). Then send the CALC:MARK:BWID command. The user range used with bandwidth search only applies to marker 1 searching for the max value. The other markers may fall outside the user range.

[See Critical Note](#)

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<num> Target value below filter peak. Choose any number between **:-500** and **500**

Examples

```
CALC:MARK:BWID -3  
calculate2:marker:bandwidth -2.513
```

Query Syntax CALCulate<cnum>:MARKer:BWIDth?
Returns the results of bandwidth search:

Return Type Numeric - Four Character values separated by commas: bandwidth, center Frequency, Q, loss.

Default -3

CALCulate<cnum>:MARKer<mkr>:COMPression:LEVel <num>

(Read-Write) Set and read the marker compression level. A compression marker must already exist. Use [CALC:MARK ON](#) and [CALC:MARK:FUNC COMP](#) to create compression markers.

[See Critical Note](#)

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<mkr> Any existing marker number from 1 to 10; if unspecified, value is set to 1.

<num> Compression level. Choose any number between: -500 dB to 500 dB

Standard gain compression values are positive.

Examples

```
CALC:MARK:COMP:LEV 1
calculate2:marker:compression:level 1.5
```

Query Syntax CALCulate<cnum>:MARKer:COMPression:LEVel?

Return Type Numeric

Default +1

CALCulate<cnum>:MARKer<mkr>:COMPression:PIN?

(Read-only) Reads the input power at the marker compression level. First send [CALC:MARK:FUNC:EXEC COMP](#) or [CALC:MARK:FUNC:TRAC ON](#)

[See Critical Note](#)

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<mkr> Any existing marker number from 1 to 10; if unspecified, value is set to 1.

Examples

```
CALC:MARK:COMP:PIN?
calculate2:marker:compression:pin?
```

Return Type Numeric

Default Not applicable

CALCulate<cnum>:MARKer:COMPression:POUT?

(Read-only) Reads the output power at the marker compression level. First send [CALC:MARK:FUNC:EXEC COMP](#) or [CALC:MARK:FUNC:TRAC ON](#)

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <mkr> Any existing marker number from 1 to 10; if unspecified, value is set to 1.

Examples

```
CALC:MARK:COMP:POUT?  
calculate2:marker:compression:pout?
```

Return Type Numeric

Default Not applicable

CALCulate<cnum>:MARKer<mkr>:COUPling[:STATe]<ON|OFF>

(Read-Write) Sets and reads the state of Coupled Markers (ON and OFF).

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <mkr> Any existing marker number from 1 to 10; if unspecified, value is set to 1.
- <ON|OFF> **OFF (0)** - Turns Coupled Markers OFF
- ON (1)** - Turns Coupled Markers ON

Examples

```
CALC:MARK:COUP ON  
calculate2:marker8:coupling off
```

Query Syntax CALCulate<cnum>:MARKer<mkr>:COUPling:[STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

CALCulate<cnum>:MARKer<mkr>:DELTA <ON|OFF>

(Read-Write) Specifies whether marker is relative to the Reference marker or absolute.

Note: The reference marker must already be turned ON with [CALC:MARK:REF:STATE](#).

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <mkr> Any existing marker number from 1 to 10; if unspecified, value is set to 1.
- <ON|OFF> **ON** (or 1) - Specified marker is a Delta marker
OFF (or 0) - Specified marker is an ABSOLUTE marker

Examples

```
CALC:MARK:DELT ON  
calculate2:marker8:delta off
```

Query Syntax CALCulate<cnum>:MARKer<mkr>:DELTA?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

CALCulate<cnum>:MARKer<mkr>:DISCrete <ON|OFF>

(Read-Write) Makes the specified marker display either a calculated value between data points (interpolated data) or the actual data points (discrete data).

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <mkr> Any existing marker number from 1 to 10; if unspecified, value is set to 1.
- <ON|OFF> **ON** (or 1) - Specified marker displays the actual data points
OFF (or 0) - Specified marker displays calculated data between the actual data points.

Examples

```
CALC:MARK:DISC ON  
calculate2:marker8:discrete off
```

Query Syntax CALCulate<cnum>:MARKer<mkr>:DISCrete?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

CALCulate<cnum>:MARKer<mkr>:DISTance <num>

(Read-Write) Set or query marker distance on a time domain trace.

The Write command moves the marker to the specified distance value. Once moved, you can [read the Y axis](#) value or [read the X-axis time](#) value. (Distance is calculated from the X-axis time value.)

The Read command reads the distance of the marker.

If the marker is set as delta, the WRITE and READ data is relative to the reference marker.

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <mkr> Any existing marker number from 1 to 10; if unspecified, value is set to 1.
- <num> Marker distance in the unit of measure specified with [CALC:TRAN:TIME:MARK:UNIT](#)

Examples

```
CALC:MARK:DIST .1  
calculate2:marker8:distance 5
```

Query Syntax CALCulate<cnum>:MARKer<mkr>:DISTance?

Return Type Numeric

Default Not Applicable

CALCulate<cnum>:MARKer<mkr>:FORMat <char>

(Read-Write) Sets the format of the data that will be returned in a marker data query CALC:MARK:Y? and the displayed value of the marker readout. The selection does not have to be the same as the measurement's display format.

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <mkr> Any marker number from 1 to 10; if unspecified, value is set to 1
- <char> Choose from:
Default - The format of the selected measurement

MLINear - Linear magnitude
MLOGarithmic - Logarithmic magnitude
IMPedance - (R+jX)
ADMittance - (G+jB)
PHASe - Phase
IMAGinary - Imaginary part (Im)
REAL - Real part (Re)
POLar - (Re, Im)
GDELay - Group Delay
LINPhase - Linear Magnitude and Phase
LOGPhase - Log Magnitude and Phase
KELVin - temperature
FAHRenheit - temperature
CELSius - - temperature

Examples

```
CALC:MARK:FORMat MLIN  
calculate2:marker8:format Character
```

Query Syntax CALCulate<cnum>:MARKer<mkr>:FORMat?

Return Type Character

Default DEFault

CALCulate<cnum>:MARKer<mkr>:FUNCtion:APEak:EXCursion <num>

(Read-Write) Sets amplitude peak excursion for the specified marker. The Excursion value determines what is considered a "peak". This command applies to marker peak searches (Next peak, Peak Right, Peak Left).

[See Critical Note](#)

Parameters

- <num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.
- <mkr> Any existing marker number from 1 to 10; if unspecified, value is set to 1.
- <num> Excursion value. Choose any number between **-500** and **500**.

Note: This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
CALC:MARK:FUNC:APE:EXC 10  
calculate2:marker8:function:apeak:excursion maximum
```

Query Syntax CALCulate<num>:MARKer<mkr>:FUNCTION:APEak:EXCursion?

Return Type Numeric

Default 3

CALCulate<num>:MARKer<mkr>:FUNCTION:APEak:THReshold <num>

(Read-Write) Sets peak threshold for the specified marker. If a peak (using the criteria set with :EXCursion) is below this reference value, it will not be considered when searching for peaks. This command applies to marker peak searches (Next peak, Peak Right, Peak Left).

[See Critical Note](#)

Parameters

- <num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.
- <mkr> Any marker number from 1 to 10; if unspecified, value is set to 1
- <num> Threshold value. Choose any number between **-500** and **500**.

Note: This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
CALC:MARK:FUNC:APE:THR -40  
calculate2:marker8:function:apeak:threshold -55
```

Query Syntax CALCulate<cnum>:MARKer<mkr>:FUNCTion:APeak:THReshold?
Return Type Numeric
Default -100

CALCulate<cnum>:MARKer<mkr>:FUNCTion:DOMain:USER[:RANGe] <range>

(Read-Write) Assigns the specified marker to a range number. The x-axis travel of the marker is constrained to the range's span. The span is specified with the [CALC:MARK:FUNC:DOM:USER:START](#) and [STOP](#) commands, unless range 0 is specified which is the full span of the analyzer.

Each channel has **16** user ranges. (Trace statistics use the same ranges.) More than one marker can use a domain range.

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <mkr> Any marker number from 1 to 10; if unspecified, value is set to 1
- User span. Choose any Integer from **0 to 16**
 - 0** is Full Span of the analyzer
 - 1 to 16** are available for user-defined x-axis span

Examples

```
CALC:MARK:FUNC:DOM:USER 1  
calculate2:marker8:function:domain:user:range 1
```

Query Syntax CALCulate<cnum>:MARKer<mkr>:FUNCTion:DOMain:USER[:RANGe]?

Returns the user span number that the specified marker is assigned to.

Return Type Numeric

Default 0 - Full Span

CALCulate<cnum>:MARKer<mkr>:FUNCTion:DOMain:USER:START <start>

(Read-Write) Sets the start of the span that the specified marker's x-axis span will be constrained to.

Use [CALC:MARK:FUNC:DOM:USER<range>](#) to set range number

Use [CALC:MARK:FUNC:DOM:USER:STOP](#) to set the stop value.

Note: If the marker is assigned to range 0 (full span), the USER:START and STOP commands generate an error. You cannot set the START and STOP values for "Full Span".

Note: This command does the same as [CALC:FUNC:DOM:USER:STAR](#)

[See Critical Note](#)

Parameters

- <cnm> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnm> is set to 1.
- <mkr> Any marker number from 1 to 10; if unspecified, value is set to 1
- <start> The analyzer's **Minimum** x-axis value

Examples

```
CALC:MARK:FUNC:DOM:USER:START 500E6  
calculate2:marker8:function:domain:user:start 1e12
```

Query Syntax CALCulate<cnm>:MARKer<mkr>:FUNCtion:DOMain:USER:START?

Return Type Numeric

Default The analyzer's **Minimum** x-axis value

CALCulate<cnm>:MARKer<mkr>:FUNCtion:DOMain:USER:STOP <stop>

(Read-Write) Sets the stop of the span that the marker's x-axis travel will be constrained to.

Use [CALC:MARK:FUNC:DOM:USER<range>](#) to set range number

Use [CALC:MARK:FUNC:DOM:USER:STARt](#) to set the stop value.

Note: If the marker is assigned to range 0 (full span), the USER:START and STOP commands generate an error. You cannot set the START and STOP values for "Full Span".

Note: This command does the same as [CALC:FUNC:DOM:USER:STOP](#)

[See Critical Note](#)

Parameters

- <num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.
- <mr> Any marker number from 1 to 10; if unspecified, value is set to 1.
- <stop> Stop value of x-axis span; Choose any number between the analyzer's **MINimum** and **MAXimum** x-axis value.

Examples

```
CALC:MARK:FUNC:DOM:USER:STOP 500e6
calculate2:marker8:function:domain1:user:stop 1e12
```

Query Syntax CALCulate<num>:MARKer<mr>:FUNCTion:DOMain:USER:STOP?

Return Type Numeric

Default The analyzer's **MAXimum** x-axis value.

CALCulate<num>:MARKer<mr>:FUNCTion:EXECute <func>

(Write-only) Immediately executes (performs) the specified search function.

[Learn more about Marker Search](#)

[See Critical Note](#)

Parameters

- <num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.
- <mr> Any marker number from 1 to 10; if unspecified, value is set to 1.
- <func> The function to be performed. Choose from:
 - **MAXimum** - finds the highest value
 - **MINimum** - finds the lowest value
 - **RPEak** - finds the next valid peak to the right
 - **LPEak** - finds the next valid peak to the left
 - **NPEak** - finds the next highest value among the valid peaks
 - **TARGET** - finds the target value to the right, wraps around to the left
 - **LTARGET** - finds the next target value to the left of the marker
 - **RTARGET** - finds the next target value to the right of the marker

- **COMP**ression - finds the compression level on a Power Swept S21 trace.

Examples

```
CALC:MARK:FUNC:EXEC MAX
calculate2:marker2:function:execute maximum
```

Query Syntax Not applicable

Default Not applicable

CALCulate<cnum>:MARKer<mkr>:FUNCtion[:SElect] <char>

(Read-Write) Sets the search function that the specified marker will perform when executed. Use [CALC:MARK:FUNC:TRAC ON](#) to automatically execute the search every sweep.

[Learn more about Marker Search](#)

[See Critical Note](#)

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<mkr> Any marker number from 1 to 10; if unspecified, value is set to 1.

<char> Marker function. Choose from:

- **MAX**imum - finds the highest value
- **MIN**imum - finds the lowest value
- **RPE**ak - finds the next valid peak to the right
- **LPE**ak - finds the next valid peak to the left
- **NPE**ak - finds the next highest value among the valid peaks
- **TAR**get - finds the target value to the right, wraps around to the left
- **LTAR**get - finds the next target value to the left of the marker
- **RTAR**get - finds the next target value to the right of the marker
- **COMP**ression - finds the compression level on a power-swept S21 trace.

Examples

```
CALC:MARK:FUNC MAX
calculate2:marker8:function:select ltarget
```

Query Syntax CALCulate<cnum>:MARKer<mkr>:FUNCtion[:SElect]?

Return Type Character

Default MAX

CALCulate<cnum>:MARKer<mkr>:TARGet[:VALue] <num>

(Read-Write) Sets the target value for the specified marker when doing Target Searches with [CALC:MARK:FUNC:SEL](#) <TARGet | RTARget | LTARget>

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <mkr> Any marker number from 1 to 10; if unspecified, value is set to 1.
- <num> Target value to search for; Units are NOT allowed.

Examples

```
CALC:MARK:TARG 2.5  
calculate2:marker8:target:value -10.3
```

Query Syntax CALCulate<cnum>:MARKer<mkr>:TARGet[:VALue]?

Return Type Numeric

Default 0

CALCulate<cnum>:MARKer<mkr>:FUNCtion:TRACking <ON | OFF>

(Read-Write) Sets the tracking capability for the specified marker. The tracking function finds the selected search function every sweep. In effect, turning Tracking ON is the same as doing a [CALC:MARK:FUNC:EXECute](#) command every sweep.

[Learn more about Marker Search](#)

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <mkr> Any marker number from 1 to 10; if unspecified, value is set to 1.
- <ON | OFF> **ON** (or 1) - The specified marker will "Track" (find) the selected function every sweep.
OFF (or 0) - The specified marker will find the selected function **only** when

the CALC:MARK:FUNC:EXECute command is sent.

Examples

```
CALC:MARK:FUNC:TRAC ON  
calculate2:marker8:function:tracking off
```

Query Syntax CALCulate<cnum>:MARKer<mkr>:FUNCTion:TRACking?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

CALCulate<cnum>:MARKer:REFerence[:STATe] <ON | OFF>

(Read-Write) Turns the reference marker (marker 10) ON or OFF. When turned OFF, existing Delta markers revert to absolute markers.

[See Critical Note](#)

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<ON | OFF> **ON** (or 1) - turns reference marker ON

OFF (or 0) - turns reference marker ON

Examples

```
CALC:MARK:REF ON  
calculate2:marker:reference:state OFF
```

Query Syntax CALCulate<cnum>:MARKer:REFerence[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

CALCulate<cnum>:MARKer:REFerence:X <num>

(Read-Write) Sets and returns the absolute x-axis value of the reference marker (marker 10).

[See Critical Note](#)

- <num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.
- <num> X-axis value. Choose any number within the operating domain of the reference marker.

Examples

```
CALC:MARK:REF:X 1e9  
calculate2:marker:reference:x 1e6
```

Query Syntax CALCulate<num>:MARKer:REFerence:X?

Return Type Numeric

Default If the first Marker, turns ON in the middle of the X-axis span. If not, turns ON at the position of the active marker.

CALCulate<num>:MARKer:REFerence:Y?

(Read-only) Returns the absolute Y-axis value of the reference marker.

[See Critical Note](#)

Parameters

- <num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

Examples

```
CALC:MARK:REF:Y?  
calculate2:marker:reference:y?
```

Return Type Character

Default Not applicable

CALCulate<num>:MARKer<mkr>:SET <char>

(Write-only) Sets the selected instrument setting to assume the value of the specified marker.

Marker Functions CENT, SPAN, START, and STOP do not work with channels that are in [CW](#) or [Segment Sweep](#) mode.

[See Critical Note](#)

Parameters

- <num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.
- <mr> Any marker number from 1 to 10; if unspecified, value is set to 1
- <chr> Choose from:
- **CENTer** - changes center frequency to the value of the marker
 - **SPAN** - changes the sweep span to the span that is defined by the delta marker and the marker that it references. Unavailable if there is no delta marker.
 - **START** - changes the start frequency to the value of the marker
 - **STOP** - changes the stop frequency to the value of the marker
 - **RLEVel** - changes the reference level to the value of the marker
 - **DELay** - changes the line length at the receiver input to the phase slope at the active marker stimulus position.
 - **CWFReq** - Sets the CW frequency to the frequency of the active marker. Does NOT change sweep type. NOT available in CW or Power Sweep. Use this argument to first set the CW Frequency to a value that is known to be within the current calibrated range, THEN set [Sweep.Type](#) to POWER or CW.

Examples

```
CALC:MARK:SET CENT  
calculate2:marker8:set span
```

Query Syntax Not Applicable

Default Not Applicable

CALCulate<num>:MARKer<mr>[:STATE] <ON|OFF>

(Read-Write) Turns the specified marker ON or OFF. **Marker 10 is the Reference Marker.** To turn all markers off, use CALC:MARK:AOFF.

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <mkr> Any marker number from 1 to 10; if unspecified, value is set to 1.
- <ON|OFF> **ON** (or 1) - turns marker ON.
OFF (or 0) - turns marker OFF.

Examples

```
CALC:MARK ON  
calculate2:marker8 on
```

Query Syntax CALCulate<cnum>:MARKer<mkr>:STATe?

Return Type Boolean (1 = ON, 0 = OFF)

Default Off

CALCulate<cnum>:MARKer<mkr>:TYPE <char>

(Read-Write) Sets the type of the specified marker.

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <mkr> Any marker number from 1 to 10; if unspecified, value is set to 1
- <char> Choose from:

- **NORMal** - a marker that stays on the assigned X-axis position unless moved or searching.
- **FIXed** - a marker that will not leave the assigned X or current Y-axis position.

Examples

```
CALC:MARK:TYPE NORM  
calculate2:marker2:type fixed
```

Query Syntax CALCulate<cnum>:MARKer<mkr>:TYPE?

Return Type Character

Default NORMal

CALCulate<cnum>:MARKer<mkr>:X <num>

(Read-Write) Sets the marker's X-axis value (frequency, power, or time). If the marker is set as delta, the SET and QUERY data is relative to the reference marker.

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <mkr> Any marker number from 1 to 10; if unspecified, value is set to 1.
- <num> Any X-axis position within the measurement span of the marker.

Note: This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
CALC:MARK:X 100Mhz  
calculate2:marker8:x maximum
```

Query Syntax CALCulate<cnum>:MARKer<mkr>:X?

Return Type Numeric

Default First Marker turns ON in the middle of the X-axis span. Subsequent markers turn ON at the position of the active marker.

CALCulate<cnum>:MARKer<mkr>:Y?

(Read-only) Reads the marker's Y-axis value. The format of the value depends on the current CALC:MARKER:FORMAT setting. If the marker is set as delta, the data is relative to the reference marker. The query always returns two numbers:

- Smith and Polar formats - (Real, Imaginary)
- LINPhase and LOGPhase - (Real, Imaginary)
- All other formats - (Value,0)

Note: To accurately read the marker Y-axis value with [trace smoothing](#) applied, the requested format must match the [displayed format](#). Otherwise, the returned value is un-smoothed data. For example, to read the smoothed marker value when measuring group delay, both the display format and the marker format must be set to (Group) Delay.

[See Critical Note](#)

Parameters

- <cnm> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnm> is set to 1.
- <mk> Any marker number from 1 to 10; if unspecified, value is set to 1.

Examples

```
CALC:MARK:Y?  
calculate2:marker3:y?
```

Query Syntax CALCulate<cnm>:MARKer<mk>:Y?

Return Type Numeric

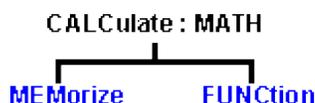
Default Not applicable

Last modified:

- | | |
|-------------|----------------------------------|
| 10-Jan-2011 | Minor edits |
| 9-Mar-2010 | Added links to PNOP and PSAT |
| 24-Feb-2009 | Replace True/False |
| 12-Feb-2009 | Added compression markers (8.50) |
| 2-Oct-2008 | Added Calc:Mark:Set CWFR (8.33) |
| 1-Oct-2007 | Added temperature formats |
| 27-Mar-2007 | Corrected Set? |
| 4-Dec-2006 | Added smoothing note to Y? |

Calculate:Math Commands

Controls math operations on the currently selected measurement and memory.



Click on a [blue](#) keyword to view the command details.

See Also

- [Example Programs](#)
- [Learn about Math Operations](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using [Calc:Par:MNUM](#) or [Calc:Par:Select](#). [Learn more](#).

CALCulate<cnum>:MATH:FUNction <char>

(Read-Write) Sets math operations on the currently selected measurement and the trace stored in memory. (There MUST be a trace stored in Memory. See [CALC:MATH MEM](#))

[See Critical Note](#)

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<char> The math operation to be applied. Choose from the following:

NORMal Trace data only

ADD Data + Memory

SUBTract Data - Memory

MULTiply Data * Memory

DIVide Data / Memory

Examples `CALC:MATH:FUNC NORM`
`calculate2:math:function subtract`

Query Syntax `CALCulate<num>:MATH:FUNction?`

Return Type Character

Default NORMal

CALCulate<num>:MATH:MEMorize

(Write-only) Puts the currently selected measurement trace into memory. (Data-> Memory).

[See Critical Note](#)

Parameters

<num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

Examples `CALC:MATH:MEM`
`calculate2:math:memorize`

Query Syntax Not applicable

Default Not applicable

Calculate:Mixer Command

CALCulate<ch>:MIXer:XAXis <char>

(Read-Write) Sets or returns the swept parameter to display on the X-axis for the selected [FCA](#) and GCX measurement.

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using [Calc:Par:MNUM](#) or [Calc:Par>Select](#). [Learn more](#).

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
- <char> Parameter to display on the X-axis. Choose from:
- INPUT** - Input frequency span
 - OUTPUT** - Output frequency span
 - LO_1** - First LO frequency span
 - LO_2** - Second LO frequency span

Examples

```
CALC:MIX:XAX INPUT  
calc2:mixer:xaxis output
```

See an example that creates, selects, and calibrates an [SMC](#) and [VMC](#) measurement using SCPI.

Query Syntax CALCulate<ch>:MIXer:XAXis?

Return Type Character

Default OUTPUT

Last Modified:

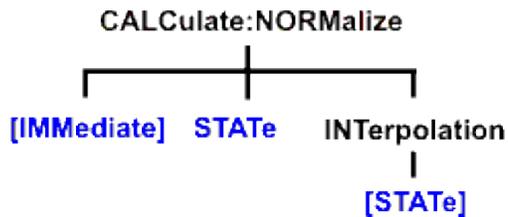
9-Sep-2010 Edit for GCX

CALCulate:NORMalize Commands

Specifies the normalization features used for a receiver power calibration.

These commands are [Superseded](#) (Sept 2004).

See the replacement commands in a new [Receiver Power Cal example](#).



Click on a [blue](#) keyword to view the command details.

See Also

- [Example Programs](#)
- [Learn about Receiver Cal](#)
- [SCPI Command Tree](#)

Save and recall your receiver power calibration (which use .CST file commands):

- [SENS:CORR:CSET:SAVE](#)
- [SENS:CORR:CSET\[:SEL\]](#)

Or use these two commands and specify either .STA or .CST file extensions:

- [MMEM:LOAD](#)
- [MMEM:STOR](#)

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using [Calc:Par:Select](#)

CALCulate<num>:NORMalize[:IMMEDIATE] **Superseded**

Note: This command is replaced with [SENS:CORR:COLL:METH RPOWer](#) and [SENS:CORR:COLL\[:ACQ\] POWer](#)

See an example of a [Receiver Power Calibration](#).

(Write only) Stores the selected measurement's data to that measurement's "divisor" buffer for use by the Normalization data processing algorithm. This command is not compatible with ratioed measurements such as S-parameters. It is intended for receiver power calibration when the selected measurement is of an unratioed power type.

[See Critical Note](#)

Parameters

<num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

Examples

```
CALC:NORM  
calculate1:normalize:immediate
```

Query Syntax Not Applicable

Default Not Applicable

CALCulate<num>:NORMalize:STATe <ON | OFF> **Superseded**

Note: This command is replaced with [SENS:CORR\[:STATe\] ON/OFF](#)

(Read-Write) Specifies whether or not normalization is applied to the measurement. Normalization is enabled only for measurements of unratioed power where it serves as a receiver power calibration.

[See Critical Note](#)

Parameters

<num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

<ON | OFF> **ON (or 1)** - normalization is applied to the measurement.

OFF (or 0) – normalization is NOT applied to the measurement.

Examples

```
CALC:NORM:STAT ON  
calculate2:normalize:state off
```

Query Syntax CALCulate<num>:NORMalize:STATe?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

CALCulate<cnum>:NORMalize:INTerpolate[:STATe] <ON | OFF> Superseded

Note: This command is replaced with [SENS:CORR:INT\[:STATe\] ON|OFF](#)

(Read-Write) Turns normalization interpolation ON or OFF. Normalization is enabled only for measurements of unratioed power, where it serves as a receiver power calibration.

[See Critical Note](#)

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<ON | OFF> **ON (or 1)** – turns interpolation ON.

OFF (or 0) – turns interpolation OFF.

Examples

```
CALC:NORM:INT ON  
calculate2:normalize:interpolate:state off
```

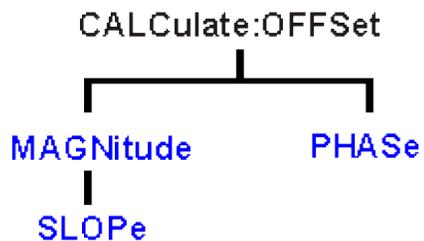
Query Syntax CALCulate<cnum>:NORMalize:INTerpolate[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

Calculate:Offset Commands

Allows the data trace magnitude and phase to be offset.



Click on a [blue](#) keyword to view the command details.

See Also

- [Example Programs](#)
- [Learn about Magnitude Offset](#)
- [Learn about Phase Offset](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using [Calc:Par:MNUM](#) or [Calc:Par>Select](#). [Learn more](#).

CALCulate<cnum>:OFFSet:MAGNitude <num>

(Read-Write) Offsets the data trace magnitude by the specified value.

To offset the data trace magnitude to a slope value that changes with frequency, use

[CALC:OFFS:MAGN:SLOP](#)

[See Critical Note](#)

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<num> Offset value in dB.

Examples

```
CALC:OFFS:MAGN:4  
calculat1:offset:magnitude -2
```

Query Syntax CALCulate<cnum>:OFFSet:MAGNitude?

Return Type Numeric

Default 0

CALCulate<cnum>:OFFSet:MAGNitude:SLOPe <num>

(Read-Write) Offsets the data trace magnitude to a value that changes linearly with frequency. The offset slope begins at 0 Hz.

[See Critical Note](#)

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<num> Offset slope value in dB/ 1GHz.

Examples

```
CALC:OFFS:MAGN:SLOP 1 'Offset slope set to 1dB/GHz  
calculat1:offset:magnitude:slope -2 'Offset slope set to -  
2dB/GHz
```

Query Syntax CALCulate<cnum>:OFFSet:MAGNitude:SLOPe?

Return Type Numeric

Default 0

CALCulate<cnum>:OFFSet:PHASe <num>[<char>]

(Read-Write) Sets the phase offset for the selected measurement.

[See Critical Note](#)

Parameters

- <num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.
- <num> Offset phase value. Choose any number between: **-360** and **360**
- <char> Units for phase. OPTIONAL. Choose either:
DEG - Degrees (default)
RAD - Radians

Examples

```
CALC:OFFS:PHAS 10  
calculate:offset:phase 20rad
```

Query Syntax CALCulate:OFFSet:PHASe?

Return Type Numeric, returned value always in degrees

Default 0 degrees

Calculate:Parameter Commands

Lists, creates, selects, and deletes measurements.

For application measurements, use [Calc:Custom commands](#).

CALCulate:PARAmeter:
CATalog
EXTended
DEFine
EXTended
DELete
ALL
MNUMber
[SElect]
MODify
EXTended
SElect
TNUMber?
WNUMber?

Click on a [blue](#) keyword to view the command details.

Red commands are [superseded](#).

See Also

- [Example Programs](#)
- [Learn about Measurement Parameters](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using [Calc:Par:MNUM](#) or [Calc:Par:Select](#). [Learn more](#).

CALCulate<cnum>:PARAmeter:CATalog? **Superseded**

Note: This command is replaced with [CALC:PAR:CAT:EXTended?](#) which lists parameters with "_" instead of "," allowing the list to be parsed easily. This command will continue to work.

(Read-only) Returns the names and parameters of existing measurements for the specified channel.

Note: For Balanced Measurements: CALC:PAR:CAT? may have an unexpected behavior. [Learn more.](#)

[See Critical Note](#)

Parameters

<cnum> Channel number of the measurements to be listed. If unspecified, <cnum> is set to 1.

Examples

```
CALC:PAR:CAT?  
calculate2:parameter:catalog?
```

Return Type String - "<measurement name>,<parameter>,[<measurement name>,<parameter>...]"

Default "CH1_S11_1,S11"

CALCulate<cnum>:PARAmeter:CATalog:EXTended?

(Read-only) Returns the names and parameters of existing measurements for the specified channel. This command lists receiver parameters with "_" such that R1,1 is reported as R1_1. This makes the returned string a true "comma-delimited" list all the time.

The returned string of this command is easily parsed and used to create measurements using the [CALC:PAR:EXT](#) command.

Parameters

<cnum> Channel number of the measurements to be listed. If unspecified, <cnum> is set to 1.

Examples

```
CALC:PAR:CAT:EXT?  
calculate2:parameter:catalog:extended?
```

Return Type String - "<measurement name>,<parameter>,[<measurement name>,<parameter>...]"

Default "CH1_S11_1,S11"

CALCulate<cnum>:PARAmeter[:DEFine] <Mname>,<param>[,port] **Superseded**

Note: This command is replaced with [CALC:PAR:DEFine:EXTended](#). This command will continue to work for up to 4-port parameters.

(Write-only) Creates a measurement but does NOT display it.

There is no limit to the number of measurements that can be created. However, there is a limit to the number of measurements that can be displayed. See [Traces, Channels, and Windows on the PNA](#).

- Use [DISP:WIND:STATE](#) to create a window if it doesn't already exist.
- Use [DISP:WIND<wnum>:TRAC<tnum>:FEED <Mname>](#) to display the measurement.

For Application Measurements see [CALC:CUST:DEF](#)

You must select the measurement (CALC<cnum>:PAR:SEL <mname>) before making additional settings.

[See Critical Note](#)

Parameters

<cnum> Channel number of the new measurement. If unspecified, value is set to 1.

<Mname> Name of the measurement. Any non-empty, unique string, enclosed in quotes.

<param> **For S-parameters:**

Any S-parameter available in the PNA

For ratioed measurements:

Any two receivers that are available in the PNA. (See the [block diagram](#) showing the receivers in YOUR PNA.)

For example: AR1 (this means A/R1)

For non-ratioed measurements:

Any receiver that is available in the PNA. (See the [block diagram](#) showing the receivers in YOUR PNA.)

For example: A

For Balanced Measurements:

First create an S-parameter measurement, then change the measurement using [CALC:FSIM:BAL](#) commands. [See an example](#).

For [Applications](#) see [CALC:CUST:DEF](#).

[port] Optional argument;

For multi-port reflection S-parameter measurements: specifies the PNA port which will provide the load for the calibration. This argument is ignored if a transmission S-parameter is specified.

For all non S-parameter measurements: specifies the source port for the measurement.

Examples

```
CALC4:PAR 'ch4_S33',S33,2 'Defines an S33 measurement with a load on port2 of the analyzer.
```

```
calculate2:parameter:define 'ch1_a', a, 1 'unratioed meas
```

```
calculate2:parameter:define 'ch1_a', ar1,1 'ratioed meas
```

Query Syntax Not Applicable; see [Calc:Par:Cat?](#)

Default Not Applicable

CALCulate<num>:PARAmeter[:DEFine]:EXTended <Mname>,<param>

Note: This command replaces [CALC:PAR:DEF](#) as it allows the creating of measurements using [external multiport testsets](#).

(Write-only) Creates a measurement but does NOT display it.

There is no limit to the number of measurements that can be created. However, there is a limit to the number of measurements that can be displayed. See [Traces, Channels, and Windows on the PNA](#).

- Use [DISP:WIND:STATe](#) to create a window if it doesn't already exist.
- Use [DISP:WIND<wnum>:TRAC<tnum>:FEED <Mname>](#) to display the measurement.

Note: For Application Measurements see [CALC:CUST:DEF](#)

You must select the measurement using [CALC:PAR:SElect](#) before making additional settings.

[See Critical Note](#)

Parameters

<num> Channel number of the new measurement. If unspecified, value is set to 1.

<Mname> **(String)** Name of the measurement. Any non-empty, unique string, enclosed in quotes.

<param> (**String**) Measurement Parameter to create. Case sensitive.

For S-parameters:

Any S-parameter available in the PNA

Single-digit port numbers CAN be separated by "_" (underscore). For example: "**S21**" or "**S2_1**"

Double-digit port numbers MUST be separated by underscore. For example: "**S10_1**"

For ratioed measurements:

Any two PNA physical receivers separated by forward slash "/" followed by comma and source port.

For example: "**A/R1, 3**"

[Learn more about ratioed measurements](#)

See a [block diagram](#) showing the receivers in YOUR PNA.

For non-ratioed measurements:

Any PNA physical receiver followed by comma and source port.

For example: "**A, 4**"

[Learn more about unratioed measurements.](#)

See the [block diagram](#) showing the receivers in YOUR PNA.

With PNA Rev 6.2, **Ratioed** and **Unratioed** measurements can also use **logical receiver notation** to refer to receivers. This notation makes it easy to refer to receivers with an [external test set](#) connected to the PNA. You do not need to know which physical receiver is used for each test port. [Learn more.](#)

For ADC measurements:

Any ADC receiver in the PNA followed by a comma, then the source port.

For example: "**AI1,2**" indicates the Analog Input1 with source port of 2.

[Learn more about ADC receiver measurements.](#)

For Balanced Measurements:

First create an S-parameter measurement, then change the measurement using [CALC:FSIM:BAL](#) "define" commands. [See an example](#).

Note: For Application Measurements see [CALC:CUST:DEF](#)

Examples

```
CALC4:PAR:EXT 'ch4_s33', 'S33' 'Defines an S33 measurement
```

```
calculate2:parameter:define:extended 'ch1_a', 'b9, 1' 'logical receiver notation for unratioed meas of test port 9 receiver with source port 1.
```

```
calculate2:parameter:define:extended 'ch1_a', 'b9/a10,1' 'logical receiver notation for ratioed meas of test port 9 receiver divided by the reference receiver for port 10 using source port 1
```

Query Syntax Not Applicable; see [Calc:Par:Cat?](#)

Default Not Applicable

CALCulate<num>:PARAmeter:DELeTe[:NAME] <Mname>

(Write-only) Deletes the specified measurement.

[See Critical Note](#)

Parameters

<num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

<Mname> String - Name of the measurement

Examples

```
CALC:PAR:DEL 'TEST'  
calculate2:parameter:delete 'test'
```

Query Syntax Not Applicable

Default Not Applicable

CALCulate:PARAmeter:DELeTe:ALL

(Write-only) Deletes all measurements on the PNA.

[See Critical Note](#)

Parameters

Examples `CALC:PAR:DEL:ALL`

Query Syntax Not Applicable

Default Not Applicable

CALCulate<cnum>:PARAmeter:MNUMber[:SElect] <n>[,fast]

(Read-Write) Sets and returns the selected measurement for the channel using the **Tr#**. Most CALC: commands require that this, or [CALC:PAR:SEL](#), be sent before a setting change is made to that measurement. Each channel can have one selected measurement.

Parameters

<cnum> Channel number of the measurement to be selected. If unspecified, <cnum> is set to 1.

<n> Numeric - Measurement number. These are the same numbers you see in the “Tr1”, “Tr2” annotation next to the parameter name on the PNA screen.

[fast] Optional. The PNA display is NOT updated. Therefore, do not use this argument when an operator is using the PNA display. Otherwise, sending this argument results in much faster sweep speeds. There is NO other reason to NOT send this argument.

Examples `CALC:PAR:MNUM 2`

`calculate2:parameter:mnumber:select 3,fast`

Query Syntax `CALCulate<cnum>:PARAmeter:MNUMber[:SElect]?`

There is NO query available to determine if the FAST argument has been set.

Return Type Numeric

Default 1 (Trace number when factory preset is performed)

CALCulate<cnum>:PARAmeter:MODIfy <param> **Superseded**

Note: This command is replaced with [CALC:PAR:MOD:EXT](#). This command will continue to work for up to 4 port parameters.

(Write-only) Modifies a standard measurement using the same arguments as [CALC:PAR:DEF](#). To modify an FCA measurement, use [CALC:CUST:MOD](#).

[See Critical Note](#)

Parameters

- <cnm> Channel number of the measurement. The selected measurement on that channel will be changed. If unspecified, <cnm> is set to 1.
- <param> Measurement parameter to change to. Use the same <param> arguments as [CALC:PAR:DEF](#).

Examples

```
SYST:PRESET
CALC:PAR:DEF "MyMeas", S11
CALC:PAR:SEL "MyMeas"
CALC:PAR:MOD AR1 'changes the selected S11 measurement to an A/R1 measurement'
```

Query Syntax Not Applicable

Default Not Applicable

CALCulate<cnm>:PARAmeter:MODify:EXTended <param>

Note: This command replaces [CALC:PAR:MOD](#) as it allows modification of measurements using [external multiport testsets](#).

(Write-only) Modifies a standard measurement using the same arguments as [CALC:PAR:DEF:EXT](#).

To modify an Application measurement, use [CALC:CUST:MOD](#).

[See Critical Note](#)

Parameters

- <cnm> Channel number of the measurement. The selected measurement on that channel will be changed. If unspecified, <cnm> is set to 1.
- <param> **(String)** New measurement parameter. Use the same <param> arguments as [CALC:PAR:DEF:EXT](#).

Examples

```
SYST:PRESET
CALC:PAR:DEF:EXT "MyMeas", "S10_1"
```

```
CALC:PAR:SEL "MyMeas"
```

```
CALC:PAR:MOD:EXT "a4b4,1" 'changes the selected S10_1  
measurement to an a4/b4 measurement with source port 1
```

Query Syntax Not Applicable

Default Not Applicable

CALCulate<cnum>:PARAmeter:SElect <Mname>[,fast]

(Read-Write) Sets the selected measurement. Most CALC: commands require that this command be sent before a setting change is made. One measurement on each channel can be selected at the same time.

- Use [CALC:PAR:MNUM](#) to select a measurement by **Tr#** number. [Learn more](#).
- To obtain a list of currently named measurements, use [CALC:PAR:CAT?](#)

Parameters

<cnum> Channel number of the measurement to be selected. If unspecified, <cnum> is set to 1.

<Mname> String - Name of the measurement. CASE-SENSITIVE. Do NOT include the parameter name that is returned with Calc:Par:Cat?

[fast] Optional. The PNA display is NOT updated. Therefore, do not use this argument when an operator is using the PNA display. Otherwise, sending this argument results in much faster sweep speeds. There is NO other reason to NOT send this argument.

Examples

```
CALC:PAR:SEL 'TEST'
```

```
calculate2:parameter:select 'test',fast
```

Query Syntax CALCulate:PARAmeter:SElect?

There is NO query available to determine if the FAST argument has been set.

Return Type String

Default "CH1_S11_1" (Trace name when factory preset is performed)

CALCulate<cnum>:PARAmeter:TNUMBER?

(Read-only) Returns the trace number of the selected trace. Select a trace using [Calc:Par:Select](#).

Parameters

<cnum> Channel number of the trace. If unspecified, <cnum> is set to 1.

Examples

```
CALC:PAR:TNUM?  
calculate2:parameter:tnumber?
```

Return Type Numeric

Default Not Applicable

CALCulate<cnum>:PARAmeter:WNUMber?

(Read-only) Returns the window number of the selected trace. Select a trace using [Calc:Par:Select](#).

Parameters

<cnum> Channel number of the selected trace. If unspecified, <cnum> is set to 1.

Examples

```
CALC:PAR:WNUM?  
calculate2:parameter:wnumber?
```

Return Type Numeric

Default Not Applicable

Last modified:

- 23-Dec-2013 Added fast to select and mnum commands
- 1-Aug-2013 Edited Calc:Par:Mod to all app meas
- 10-Jul-2013 Added note about using EXT commands together.
- 7-Mar-2012 Added 'for apps, use...'
- 26-May-2011 Added TNUM and WNUM
- 10-Feb-2011 Removed defaults from MNUM and CPS
- 31-Oct-2008 Added Mnum select (8.33)
- 19-Apr-2007 Added ADC meas
- 9/12/06 New Extended commands.

Calculate:RData? Command

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using [Calc:Par:MNUM](#) or [Calc:Par:Select](#). [Learn more](#).

CALCulate<cnum>:RDATA? <char>

(Read-only) Returns receiver data for the selected measurement. To query measurement data, see [CALC:DATA?](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <char> Choose from any physical receiver in the PNA.

For example: "A"

Also, **REF** - returns data for either R1 or R2 data depending on the source port of the selected measurement.

See the [block diagram](#) showing the receivers in YOUR PNA.

Note: Logical receiver notation is NOT allowed with this command. [Learn more](#).

Example

```
GPIB.Write "INITiate:CONTinuous OFF"  
GPIB.Write "INITiate:IMMediate;*wai"  
GPIB.Write "CALCulate:RDATA? A"
```

```
GPIB.Write "CALCulate:RDATA? REF"
```

Return Type Depends on [FORM:DATA](#) - Two numbers per data point

Default Not Applicable

Notes:

Generally when you query the analyzer for data, you expect that the number of data values returned will be consistent with the number of points in the sweep.

However, if you query **receiver** data while the instrument is sweeping, the returned values may contain zeros. For example, if your request for receiver data is handled on the 45th point of a 201 point sweep, the first 45 values will be valid data, and the remainder will contain complex zero.

This can be avoided by synchronizing this request with the end of a sweep or putting the channel in hold mode.

[Learn about Unratioed Measurements](#)

Calculate:Smoothing Commands

Controls point-to-point smoothing. Smoothing is a noise reduction technique that averages adjacent data points in a measurement trace. Choose the amount of smoothing by specifying either the number of points or the aperture. Smoothing is not the same as CALC:AVERage which averages each data point over a number of sweeps.



Click on a [blue](#) keyword to view the command details.

See Also

- [Example Programs](#)
- [Learn about Smoothing](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using [Calc:Par:MNUM](#) or [Calc:Par>Select](#). [Learn more](#).

CALCulate<cnum>:SMOothing:APERture <num>

(Read-Write) Sets the amount of smoothing as a percentage of the number of data points in the channel.

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <num> Percentage value. Choose any number between:
1 and **25**

Examples

```
CALC:SMO:APER 2  
calculate2:smoothing:aperture 20.7
```

Query Syntax CALCulate<cnum>:SMOothing:APERture?

Return Type Numeric

Default 1.5

CALCulate<cnum>:SMOothing:POINts <num>

(Read-Write) Sets the number of adjacent data points to average.

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <num> Number of points from 1 point to maximum of 25% of data points in the channel. For example: if number of points in a data trace = 401, the maximum value for points = 100. The points value is always rounded to the closest odd number.

Examples

```
CALC:SMO:POIN 50  
calculate2:smoothing:points 21
```

Query Syntax CALCulate<cnum>:SMOothing:POINts?

Return Type Numeric

Default 3

CALCulate<cnum>:SMOothing[:STATe] <ON | OFF>

(Read-Write) Turns data smoothing ON or OFF.

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <ON | OFF> **ON** (or 1) - turns smoothing ON.
OFF (or 0) - turns smoothing OFF.

Examples

```
CALC:SMO ON  
calculate2:smoothing:state off
```

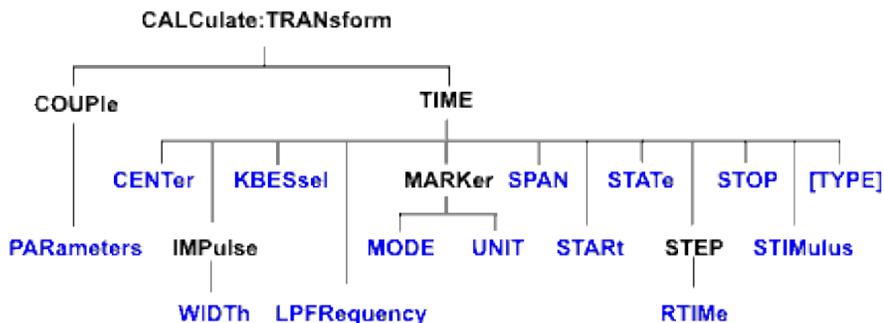
Query Syntax CALCulate<cnum>:SMOothing[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

Calculate:Transform Commands

Specifies the settings for time domain transform.



Click on a [blue](#) keyword to view the command details.

See Also

- [Example Programs](#)
- [Learn about Time Domain](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using [Calc:Par:MNUM](#) or [Calc:Par:Select](#).

CALCulate<num>:TRANSform:COUPlE:PARAmeters <num>

(Read-Write) Specifies the time domain transform parameters to be coupled. The settings for those parameters will be copied from the selected measurement to all other measurements on the channel.

- To turn coupling ON and OFF, use [SENS:COUP:PAR](#)
- To specify Gating parameters to couple, use [CALC:FILT:COUP:PAR](#)

Learn more about [Time Domain Trace Coupling](#)

[See Critical Note](#)

Parameters

<num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

<num> (Numeric) Parameters to couple. To specify more than one parameter, add the

numbers.

1 - Transform Stimulus (Start, Stop, Center, and Span TIME settings.)

2 - Transform State (ON / OFF)

4 - Transform Window (Kaiser Beta / Impulse Width)

8 - Transform Mode (Low Pass Impulse, Low Pass Step, Band Pass)

16 - Transform Distance Marker Units

Examples

```
'To couple all parameters:
```

```
CALC:TRAN:COUP:PAR 31
```

```
'To couple Stimulus and Mode:
```

```
calculate2:transform:couple:parameters 9
```

Query Syntax CALCulate<cnum>:TRANSform:COUPlE:PARAmeters?

Return Type Numeric

Default 29 (All parameters except 2 - Transform State)

CALCulate<cnum>:TRANSform:TIME:CENTer <num>

(Read-Write) Sets the center time for time domain measurements.

[See Critical Note](#)

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<num> Center time in seconds; any number between:
 $\pm (\text{number of points}-1) / \text{frequency span}$

Note: This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
CALC:TRAN:TIME:CENT 1e-8
```

```
calculate2:transform:time:center 15 ps
```

Query Syntax CALCulate<cnum>:TRANSform:TIME:CENTer?

Return Type Numeric

Default 0

CALCulate<cnum>:TRANSform:TIME:IMPulse:WIDTh <num>

(Read-Write) Sets the impulse width for the transform window.

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <num> Impulse width in seconds; Choose any number between:
.6 / frequency span and **1.39 / frequency span**

Examples

```
CALC:TRAN:TIME:IMP:WIDTh 10  
calculate2:transform:time:impulse:width 13
```

Query Syntax CALCulate<cnum>:TRANSform:TIME:IMPulse:WIDTh?

Return Type Numeric

Default .98 / Default Span

CALCulate<cnum>:TRANSform:TIME:KBESsel <num>

(Read-Write) Sets the parametric window for the Kaiser Bessel window.

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <num> Window width for Kaiser Bessel in seconds; Choose any number between:
0.0 and **13.0**

Examples

```
CALC:TRAN:TIME:KBES 10  
calculate2:transform:time:kbessel 13
```

Query Syntax CALCulate<cnum>:TRANSform:TIME:KBESsel?

Return Type Numeric

Default 6

CALCulate<cnum>:TRANSform:TIME:LPFREQuency

(Write-only) Sets the start frequencies in LowPass Mode.

[See Critical Note](#)

Parameters

<num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

Examples

```
CALC:TRAN:TIME:LPFR  
calculate2:transform:time:lpfrequency
```

Query Syntax Not applicable

Default Not applicable

CALCulate<num>:TRANSform:TIME:MARKer:MODE <char>

(Read-Write) Specifies the measurement type in order to determine the correct marker distance.

- Select Auto for S-Parameter measurements.
- Select Reflection or Transmission for arbitrary ratio or unratiod measurements.

This setting affects the display of ALL markers for only the ACTIVE measurement.

Learn more about [Distance Markers](#).

[See Critical Note](#)

Parameters

<num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.

<char> Choose from:

AUTO If the active measurement is an S-Parameter, automatically chooses reflection or transmission. If non S-Parameter measurements, reflection is chosen.

REFlection Displays the distance from the source to the receiver divided by two (to compensate for the return trip.)

TRANsmission Displays the distance from the source to the receiver.

Examples

```
CALC:TRAN:TIME:MARK:MODE REFL  
calculate2:transform:time:marker:mode auto
```

Query Syntax CALCulate<num>:TRANSform:TIME:MARKer:MODE?

Return Type Character

Default Auto

CALCulate<cnum>:TRANSform:TIME:MARKer:UNIT <char>

(Read-Write) Specifies the unit of measure for the display of marker distance values. This settings affects the display of ALL markers for only the ACTIVE measurement (unless Distance Maker Units are coupled using [CALC:TRAN:COUP:PAR](#)).

Learn more about [Distance Markers](#).

[See Critical Note](#)

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<char> Choose from:

METRs

FEET

INCHes

Examples

```
CALC:TRAN:TIME:MARK:UNIT INCH  
calculate2:transform:time:marker:unit feet
```

Query Syntax CALCulate<cnum>:TRANSform:TIME:MARKer:UNIT?

Return Type Character

Default METRs

CALCulate<cnum>:TRANSform:TIME:SPAN <num>

(Read-Write) Sets the span time for time domain measurements.

[See Critical Note](#)

Parameters

- <num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.
- <num> Span time in seconds; any number between:
0 and $2 * [(number\ of\ points - 1) / frequency\ span]$

Note: This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
CALC:TRAN:TIME:SPAN 1e-8  
calculate2:transform:time:span maximum
```

Query Syntax CALCulate<num>:TRANSform:TIME:SPAN?

Return Type Numeric

Default 20 ns

CALCulate<num>:TRANSform:TIME:STARt <num>

(Read-Write) Sets the start time for time domain measurements.

[See Critical Note](#)

Parameters

- <num> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <num> is set to 1.
- <num> Start time in seconds; any number between:
 $\pm (number\ of\ points - 1) / frequency\ span$

Note: This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
CALC:TRAN:TIME:STAR 1e-8  
calculate2:transform:time:start minimum
```

Query Syntax CALCulate<num>:TRANSform:TIME:STARt?

Return Type Numeric

Default -10 ns

CALCulate<cnum>:TRANSform:TIME:STATE <ON | OFF>

(Read-Write) Turns the time domain transform capability ON or OFF.

[See Critical Note](#)

Note: [Sweep type](#) must be set to Linear Frequency in order to use Time Domain Transform.

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<ON|OFF> **ON** (or 1) - turns time domain ON.
OFF (or 0) - turns time domain OFF.

Examples

```
CALC:TRAN:TIME:STAT ON  
calculate2:transform:time:state off
```

Query Syntax CALCulate<cnum>:TRANSform:TIME:STATE?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

CALCulate<cnum>:TRANSform:TIME:STOP <num>

(Read-Write) Sets the stop time for time domain measurements.

[See Critical Note](#)

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<num> Stop time in seconds; any number between:
 $\pm (\text{number of points}-1) / \text{frequency span}$

Note: This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
CALC:TRAN:TIME:STOP 1e-8  
calculate2:transform:time:stop maximum
```

Query Syntax CALCulate<cnum>:TRANSform:TIME:STOP?

Return Type Numeric

Default 10 ns

CALCulate<cnum>:TRANSform:TIME:STEP:RTIME <num>

(Read-Write) Sets the step rise time for the transform window.

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <num> Rise time in seconds; Choose any number between:
.45 / frequency span and **1.48 / frequency span**

Examples

```
CALC:TRAN:TIME:STEP:RTIM 1e-8  
calculate2:transform:time:step:rtime 15 ps
```

Query Syntax CALCulate<cnum>:TRANSform:TIME:STEP:RTIME?

Return Type Numeric

Default .99 / Default Span

CALCulate<cnum>:TRANSform:TIME:STIMulus <char>

(Read-Write) Sets the type of simulated stimulus that will be incident on the DUT.

[See Critical Note](#)

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <char> Choose from:
STEP - simulates a step DUT stimulus
IMPulse - simulates a pulse DUT stimulus

STEP can ONLY be used when [CALC:TRAN:TIME:TYPE](#) is set to LPASs (Lowpass). (STEP **cannot** be used with TYPE = BPASs.)

:STIM STEP will set :TYPE to **LPASs**

:TYPE BPASs will set :STIM to **IMPulse**

Examples

```
CALC:TRAN:TIME:STIM STEP  
calculate2:transform:time:stimulus impulse
```

Query Syntax CALCulate<cnum>:TRANSform:TIME:STIMulus?

Return Type Character

CALCulate<cnum>:TRANSform:TIME[:TYPE] <char>

(Read-Write) Sets the type of time domain measurement.

[See Critical Note](#)

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<char> Type of measurement. Choose from:
LPASs - Lowpass; Must also send [CALC:TRAN:TIME:LPFRequency](#) before calibrating.

BPASs - Bandpass;

BPASs can **only** be used when [CALC:TRAN:TIME:STIM](#) is set to IMPulse.
(BPASs **cannot** be used with :STIM = STEP)

:STIM **STEP** will set :TYPE to **LPASs**

:TYPE **BPASs** will set :STIM to **IMPulse**

Examples

```
CALC:TRAN:TIME LPAS  
calculate2:transform:time:type bpas
```

Query Syntax CALCulate<cnum>:TRANSform:TIME[:TYPE]?

Return Type Character

Default BPAS

X Values Command

CALCulate<cnum>:X[:VALues]?

(Read-only) Returns the stimulus values for the selected measurement in the current units. You can select one measurement for each channel using [Calc:Par:MNUM](#) or [Calc:Par:Select](#). [Learn more](#).

This command can be used for all Measurement Classes.

Note: To avoid frequency rounding errors, specify [FORM:DATA](#) <Real,64> or <ASCIi, 0>

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1.

Examples

```
1. Calc:Par:Sel "MyGCATrace"  
2. CALC:X?
```

Return Type Depends on [FORM:DATA](#) command

Default Not applicable

Last Modified:

10-Sep-2012	Added Note
12-Aug-2009	MX New topic

CalPod Commands

The following commands are sent as a string argument from:

[CONTRol:CALPod:COMMand <string>](#)

CALPod

DISable

ENABle

HIDE

INITialize

| **ACTive**

| **ALL**

LAUNCh

RECorrect

| **ACTive**

| **ALL**

SHOW

STATe

TEMP?

Click on a **blue** keyword to view the command details.

In addition to the above Calpod commands, the following IEE 488 Common Commands can also be sent as a string argument:

- ***CLS** - Clears all errors and event data from the error/event queue.
- ***IDN?** - Returns the instrument identification information.
- ***OPC?** - Operation complete query. This query immediately returns a value, independent of whether or not the operation is complete. A return value of 0 indicates the operation is not complete. A value of +1 indicates the operation is complete. Typically this command is used in a loop with a 0.25 second delay when waiting for an operation to complete.

- ***TST?** - Performs a communication test on all the currently enabled Calpods. 0 = Test failed on one or more enabled Calpods. 1 = All enabled Calpods working.
- **SYSTEM:ERROR?** - Queries the Event/Error queue and returns the most recent error element.

Important Notes

- ALL commands on this page are sent as a string argument from: [CONTrol:CALPod:COMMand <string>](#)
- Use single quotes ONLY (NOT double quotes) for the CONT:CALP:COMM string arguments.
- Sending queries requires TWO question marks. See following note as example.
- To read errors with the commands on this page, use the Calpod query:
`CONT:CALP:COMM? 'SYSTEM:ERROR?'`
- ALL queries return strings.

See Also

- [Example Programs](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

CALPod:DISable <port>

(Write-only) Unassign Calpod serial number from the specified PNA port.

[See important notes.](#)

Parameters

<port> PNA port number to un-assign.

Examples `CONT:CALP:COMM 'CALP:DIS 2'`

Query Syntax Not Applicable

Default Not Applicable

CALPod:ENABLE <port>,<sn>

(Write-read) Assign or return the Calpod serial number for the specified PNA port. If a Calpod module is already assigned to the specified PNA port, this assignment will replace the existing assignment.

[See important notes.](#)

Parameters

<port> PNA port number to be assigned the Calpod serial number.

<sn> Calpod serial number.

Examples

```
CONT:CALP:COMM 'CALP:ENAB 2, 0001234' 'WRITE
```

```
CONT:CALP:COMM? 'CALP:ENAB? 2' 'READ
```

Query Syntax CONTrol:CALPod:COMMand? 'CALPod:ENABle? <port>'

Return Type String

Default Not Applicable

CALPod:HIDE

(Write-only) Hides the Calpod setup dialog.

[See important notes.](#)

Parameters None

Examples

```
CONT:CALP:COMM 'CALP:HIDE'
```

Query Syntax Not Applicable

Default Not Applicable

CALPod:INITialize:ACTive

(Write-only) Performs the initialize process for the active (selected) channel. Select a channel using [Calc:Par:Select](#).

[See important notes.](#)

Parameters None

Examples

```
CONT:CALP:COMM 'CALP:INIT:ACT'
```

Query Syntax Not Applicable

Default Not Applicable

CALPod:INITialize:ALL

(Write-only) Performs the initialize ALL channels process.

[See important notes.](#)

Parameters	None
Examples	<code>CONT:CALP:COMM 'CALP:INIT:ALL'</code>
Query Syntax	Not Applicable
Default	Not Applicable

CALPod:LAUNCh

(Write-only) Starts the Calpod software. The Calpod software can be started using this (Launch) command or by activating the Calpod user interface. Once the Calpod software is started it remains active until the PNA application is terminated.

Send this command first in your program, then wait a couple seconds while the software starts before sending the next command.

[See important notes.](#)

Parameters	None
Examples	<code>CONT:CALP:COMM 'CALP:LAUN'</code> <code>wait 3</code>
Query Syntax	Not Applicable
Default	Not Applicable

CALPod:RECorrect:ACTive

(Write-only) Performs the recorrect process for the active (selected) channel. Select a channel using [Calc:Par:Select](#).

[See important notes.](#)

Parameters	None
Examples	<code>CONT:CALP:COMM 'CALP:REC:ACT'</code>
Query Syntax	Not Applicable
Default	Not Applicable

CALPod:RECorrect:ALL

(Write-only) Performs the recorrect process for ALL channels.

[See important notes.](#)

Parameters None

Examples `CONT:CALP:COMM 'CALP:REC:ALL'`

Query Syntax Not Applicable

Default Not Applicable

CALPod:SHOW

(Write-only) Shows the Calpod setup dialog.

[See important notes.](#)

Parameters None

Examples `CONT:CALP:COMM 'CALP:SHOW'`

Query Syntax Not Applicable

Default Not Applicable

CALPod:STATE <sn>,<state>

(Write-only) Sets the specified Calpod module to specified impedance state.

[See important notes.](#)

Parameters

<sn> Serial number of the Calpod module. When set to **1**, all modules are set to the specified state.

<state> Impedance state. Not case sensitive. Choose from:

Short, Open, Load, or Thru

Examples `CONT:CALP:COMM 'CALPod:STATE 0001234,thru'`

Query Syntax Not Applicable

Default Thru

CALPod:TEMP? <sn>

(Read-only) Returns the temperature of the specified Calpod module in degrees Celsius.

[See important notes.](#)

Parameters

<sn> Serial number of the Calpod module.

Examples

```
CONT:CALP:COMM? 'CALPod:TEMP? 0001234'
```

Query Syntax Not Applicable

Return Type String

Default Not Applicable

Last Modified:

11-Apr-2013	Fixed typo (BH) and removed note for launch
27-Sep-2012	Updated the Launch command (LH)
14-Sep-2012	Removed links from IEEE commands (BH)
25-Jun-2012	Edit OPC? (LH)
4-Jan-2011	New topic

Control Commands

Specifies the settings to remotely control the rear panel connectors, an external test set, Calpod modules, and ECal Module state.

CONTrol

[AUXiliary - More Commands](#)

[CALPod:COMMand](#)

CHANnel:INTerface:CONTrol:

| [CONFig:RECall](#)

| [\[STATe\]](#)

ECAL:MODule:

| **PATH:**

| [COUNT?](#)

| [STATE](#)

| **STATE**

[EXTernal:TESTset - More Commands](#)

[HANDler - More Commands](#)

[NOISe:SOURce\[:STATe\]](#)

SIGNal:

| **TRIGger**

| [ATBA](#)

| [OUTP](#)

Click on a [blue](#) keyword to view the command details.

Red command is superseded.

See Also

- [Example Programs](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)
- See a pinout and detailed description of the rear panel connectors:
 - [External Test Set IO connector](#)
 - [Material Handler IO connector](#)

CONTrol:CALPod:COMMand <string>

(Write-Read) Sends commands that control a Calpod module. Reads query versions Calpod commands.

See [ALL Calpod commands](#).

[Learn more about Calpod.](#)

Parameters

<string> Calpod command. See [ALL Calpod commands](#) that can be used in this string.

Write Example

```
CONT:CALP:COMM 'CALP:INIT:ACT'
```

'Enclose all strings in SINGLE quotes (NOT double quotes)

Query Syntax

CONTrol:CALPod:COMMand? <string>

Relevant only for query strings.

Read Example

```
CONT:CALP:COMM? '*OPC?'
```

'returns 0 if the calpod software is currently processing an operation

'returns 1 if operations are complete

Return Type

String

Default

Not Applicable

CONTrol:CHANnel:INTERface:CONTrol:CONFig:RECall[:STATe] <string>

(Write-only) Recalls an Interface Control configuration file. [Learn more about Interface Control.](#)

Parameters

<string> File name and extension (.xml) of the configuration file to recall. Files are typically stored in the default folder "C:/Program Files/Agilent/Network Analyzer/Documents". To recall from a different folder, specify the full path name.

Examples

```
CONT:CHAN:INT:CONT:CONF:REC 'MyConfigFile.xml'  
  
control:channel:interface:control:config:recall:state  
'c:\users\public\documents\Network  
analyzer\Documents\MyFile.xml'
```

Query Syntax Not Applicable

Default Not Applicable

CONTrol:CHANnel:INTerface:CONTrol[:STATe] <bool>

(Read-Write) Enables and disables ALL Interface Control settings. To send data, the individual interfaces must also be enabled. [Learn more about Interface Control.](#)

Parameters

<bool> Boolean

OFF (0) - Interface Control is disabled;NO control data is sent.

ON (1) - Interface Control is enabled.

Examples

```
CONT:CHAN:INT:CONT 1  
  
control:channel:interface:control:state 0
```

Query Syntax CONTrol:CHANnel:INTerface:CONTrol[:STATe]?

Return Type Boolean

Default OFF (0)

CONTrol:ECAL:MODule<num>:PATH:COUNT? <name>

(Read-only) Returns the number of unique states that exist for the specified path name on the selected ECal module.

This command performs exactly the same function as [SENS:CORR:CKIT:ECAL:PATH:COUNT?](#)

Use the [CONT:ECAL:MOD:PATH:STAT](#) command to set the module into one of those states.

Use [SENS:CORR:CKIT:ECAL:PATH:DATA?](#) to read the data for a state.

Parameters

[num] Optional argument. USB number of the ECal module. If unspecified (only one ECal module is connected to the USB), <num> is set to 1. If two or more modules are connected, use [SENS:CORR:CKIT:ECAL:LIST?](#) to determine how many, and [SENS:CORR:CKIT:ECAL:INF?](#) to verify their identities.

<name> Name of the path for which to read number of states. Choose from:

Reflection paths

- **A**
- **B**
- **C** (4-port modules)
- **D** (4-port modules)

Transmission paths

- **AB**
- **AC** (4-port modules)
- **AD** (4-port modules)
- **BC** (4-port modules)
- **BD** (4-port modules)
- **CD** (4-port modules)

Note: For each transmission path, the first of the available states is the through state, the second is the confidence (attenuator) state.

Examples

```
CONT:ECAL:MOD:PATH:COUNT? A
control:ecal:module2:path:count? cd
See example program
```

Return Type Integer

Default Not Applicable

CONTrol:ECAL:MODule<num>:PATH:STATe <path>, <stateNum>

(Write-only) Sets the internal state of the selected ECAL module. This command supersedes [CONT:ECAL:MOD:STAT](#).

- Use [CONT:ECAL:MOD:PATH:COUN?](#) to read the number of unique states that exist for the specified path name on the module.
- Use [SENS:CORR:CKIT:ECAL:PATH:DATA?](#) to read the data for a state (from the module memory) corresponding to the stimulus values of a channel.

Parameters

- [num] Optional argument. USB number of the ECal module. If unspecified (only one ECal module is connected to the USB), <num> is set to 1. If two or more modules are connected, use [SENS:CORR:CKIT:ECAL:LIST?](#) to determine how many, and [SENS:CORR:CKIT:ECAL:INF?](#) to verify their identities.
- <path> Path name for which to set a state.

Note: The impedance paths are not independent. For example, changing the impedance presented on path A will cause a change to the impedance on path B.

Choose from:

Reflection paths

- **A**
- **B**
- **C** (4-port modules)
- **D** (4-port modules)

Transmission paths

- **AB**
- **AC** (4-port modules)
- **AD** (4-port modules)
- **BC** (4-port modules)
- **BD** (4-port modules)
- **CD** (4-port modules)

<stateNum> Number of the state to set. Refer to the following table to associate the

<stateNum> with a state in your ECal module.

In addition, [CONT:ECAL:MOD:PATH:COUNt?](#) returns the number of states in the specified ECal module.

<stateNum>	N4432A and N4433A States	N4431A States	N469x States**	8509x States
One-Port Reflection States				
1	Open	Open	Impedance 1	Open
2	Short	Short	Impedance 2	Short
3	Impedance 1	Impedance 1	Impedance 3	Impedance 1
4	Impedance 2	Impedance 2	Impedance 4	Impedance 2
5			Impedance 5	
6			Impedance 6	
7			Impedance 7	
Two-Port Transmission States				
1	Thru	Thru	Thru	Thru
2	Confidence	Confidence	Confidence	Confidence

** The following modules have only FOUR Impedance states (1, 2, 3, 4):
N4690B ,N4691B ,N4692A ,N4696B.

Examples

```
CONT:ECAL:MOD:PATH:STATe A,5
control:ecal:module2:state BC,1
See example program
```

Query Syntax Not Applicable

Default Not Applicable

CONTrol:ECAL:MODule<num>:STATe <value> Superseded

This command is replaced with [CONT:ECAL:MOD:PATH:STATE](#).

(Write-only) Sets the internal state of the selected ECAL module.

Parameters

- [num] Optional argument. USB number of the ECal module. If unspecified (only one ECal module is connected to the USB), <num> is set to 1. If two or more modules are connected, use [SENS:CORR:COLL:CKIT:INF?](#) to verify their identity.
- <value> Integer code for switching the module. The following are codes for Agilent ECal modules.

Agilent 8509x Modules		
State	Port A	Port B
Open	0	0
Short	43	43
Load	33	33
Mismatch	4	16
Thru	84	
Confidence	20	

Agilent N469x Modules		
State	Port A	Port B
Open	36	33
Short	39	45
Load	37	37
Mismatch (Offset short)	53	53
Impedance 5 (Offset open)	5	5
Impedance 6 (Offset short)	21	21
Impedance 7 (Offset short)	38	41
Thru	42	
Confidence	40	

Agilent N4431A Modules				
State	Port A	Port B	Port C	Port D
Open	-1398	-1384	-2774	-2654
Short	-1350	-1381	-2582	-2642
Load	26985	-26986	-26986	26985
Mismatch	-26986	26985	26985	-26986
Path		Thru	Confidence	
AB Path		-2590	-598	
AC Path		-4011	85	
AD Path		-2517	16042	
BC Path		-1650	-598	
BD Path		-4011	85	
CD Path		-1352	16042	

Agilent N4432A and N4433A Modules				
State	Port A	Port B	Port C	Port D
Open	-6971	-11835	-14895	-14876
Short	-14395	-12859	-14899	-14905
Load	-14907	-14907	-14907	-14907
Offset Short	-9787	-6459	-14874	-14887
Path		Thru	Confidence	
AB Path		13765	30069	
AC Path		-10519	-2327	
AD Path		-10538	-2346	
BC Path		-5655	-1559	
BD Path		-5674	-1578	
CD Path		-15051	30069	

Examples

```
CONT:ECAL:MOD:STAT 36
control:ecal:module2:state 38
```

Query Syntax Not Applicable

Default Not Applicable

CONTROL:NOISE:SOURCE[:STATE] <bool>

(Read-Write) Set and read the noise source [\(28V\)](#) ON and OFF.

Parameters

<bool> Boolean

OFF (0) - Noise Source OFF

ON (1) - Noise Source ON

Examples

```
CONT:NOIS:SOUR 1  
control:noise:source:state 0
```

Query Syntax CONTrol:NOISe:SOURce[:STATE]?

Return Type Boolean

Default For PNA models with a [Noise Figure option](#) (028/029/H29), the 28V line is ON at application start and after a preset. The ON/OFF state is also available from a PNA softkey menu.

For PNA models WITHOUT a Noise Figure option (028/029/H29), the 28V line is OFF at application start and its state is not affected by a preset. The ON/OFF state is NOT available from a PNA softkey menu.

CONTrol:SIGNal <conn>,<char>

(Read-Write) Configures external triggering in the PNA.

Note: To configure external triggering in the models, use the [Trigger](#) commands.

- To control BNC1 and BNC2 with this command, then you **MUST** have [TRIG:PREF:AIGLobal = ON](#). [Learn more](#)
- [Trigger:Sequence:Source](#) is automatically set to External when **CONTrol:SIGNal** is sent.
- Edge triggering is only available on some Microwave PNA models.
- For more information, see [External Triggering](#) in the PNA.

Parameters

<conn> Rear Panel connector to send or receive trigger signals. Choose from:

BNC1 Trigger IN from rear-panel Trigger IN BNC connector

Note: Only one of the input connectors is active at a time. When a command is sent to one, the PNA automatically makes the other INACTIVE.

BNC2 Trigger OUT to rear-panel Trigger OUT BNC connector.

MATHtrigger - Trigger IN from rear-panel [Material Handler connector Pin 18](#)

RDY - Ready for trigger OUT.

- PNA-X: [Meas Trig RDY](#)
- PNA-L: [Handler I/O p21 \(Some models\)](#)

<char> **INACTIVE** - Disables the specified connector <conn>.

Choose from ONLY the following when <conn> is set to **BNC1** or **AUXT** or **MATHtrigger**:

- **TIENEGATIVE** - (Trigger In Edge Negative) - Triggers the PNA when receiving a negative going signal
- **TIEPOSITIVE** - (Trigger In Edge Positive) - Triggers the PNA when receiving a positive going signal
- **TILLOW** - (Trigger In Level Low) - Triggers the PNA when receiving a low level signal
- **TILHIGH** - (Trigger In Level High) - Triggers the PNA when receiving a High-level signal

Choose from ONLY the following when <conn> is set to **BNC2**:

Use [CONTrol:SIGNal:TRIGger:OUTP](#) to enable the BNC2 output.

The following selections send a positive or negative pulse before or after each trigger acquisition. This normally occurs each sweep unless a channel is in [point trigger](#) mode.

- **TOPPAFTER** - (Trigger Out Pulse Positive After) - Sends a POSITIVE going TTL pulse at the END of each trigger acquisition.
- **TOPPBEFORE** - (Trigger Out Pulse Positive Before) - Sends a POSITIVE going TTL pulse at the START of each trigger acquisition.
- **TOPNAFTER** - (Trigger Out Pulse Negative After) - Sends a NEGATIVE going TTL pulse at the END of each trigger acquisition.
- **TOPNBEFORE** - (Trigger Out Pulse Negative Before) - Sends a NEGATIVE going TTL pulse at the START of each trigger acquisition.

Choose from ONLY the following when <conn> is set to **RDY**:

- **LOW** Outputs a TTL low when the PNA is ready for trigger. (Default setting)
- **HIGH** Outputs a TTL high when the PNA is ready for trigger.

Examples

```
CONT:SIGN BNC1, TIENEGATIVE
control:signal bnc2, toppbefore

CONT:SIGN RDY, LOW
```

Query Syntax CONTrol:SIGNal? <conn>

In addition to the arguments listed above, the following is also a possible returned value:

NAVAILABLE - This feature is not available on this PNA

Return Type Character

Default At Preset:

BNC1 = INACTIVE
 BNC2 = INACTIVE
 AUXT = TILHIGH

When [Output is enabled](#):

BNC1 = INACTIVE
 BNC2 = TOPPAFTER
 AUXT = TILHIGH

CONTrol:SIGNal:TRIGger:ATBA <bool>

(Read-Write) Accept Trigger Before Armed Determines what happens to an EDGE trigger signal if it occurs before the PNA is ready to be triggered. (LEVEL trigger signals are always ignored.) For more information, see [External triggering](#).

Parameters

<bool> Boolean

OFF (0) - A trigger signal is ignored if it occurs before the PNA is ready to be triggered.

ON (1) - A trigger signal is remembered and then used when the PNA becomes armed (ready to be triggered). The PNA remembers only one trigger signal.

Examples

```
CONT:SIGN:TRIG:ATBA 0
control:signal:trigger:atba ON
```

Query Syntax CONTrol:SIGNal:TRIGger:ATBA?

Return Type Boolean

Default OFF

CONTrol:SIGNal:TRIGger:OUTP <bool>

(Read-Write) Output Enabled The PNA can be enabled to send trigger signals out the rear-panel TRIGGER OUT BNC connector. Use [CONTrol:SIGNal](#) to configure for output triggers.

Note: To configure external triggering in the models, use the [Trigger](#) commands.

For more information, see [External triggering](#).

Parameters

<bool> Boolean

OFF (0) - PNA does NOT output trigger signals.

ON (1) - PNA DOES output trigger signals.

Examples

```
CONT:SIGN:TRIG:OUTP 1
```

```
control:signal:trigger:outp OFF
```

Query Syntax CONTrol:SIGNal:TRIGger:OUTP?

Return Type Boolean

Default OFF

Last Modified:

6-Feb-2014	Replaced Aux link
25-Nov-2013	Changed file location
14-Sep-2012	Added calpod read example
12-Apr-2012	Removed links to old models
7-Jan-2011	Added Calpod
3-Nov-2010	Noise source on/off for all models
24-Feb-2009	Replaced True/False
14-Mar-2008	Added RDY argument to CONT:SIGN
25-Feb-2008	Clarified CONT:SIGN
30-Jan-2008	Added ECal states note
22-Aug-2007	Added noise command (8.0)
18-Jan-2007	Fixed count? example

CSET:Fixture Commands

Manages several aspects of Cal Sets.

CSET:

| [CATalog?](#)

| [DALL](#)

| [DATE?](#)

| [DELeTe](#)

| [EXISts?](#)

| [FIXTure:](#)

 | [DEEMbed](#)

 | [EMBed](#)

| [TIME?](#)

Click on a [blue](#) keyword to view the command details.

Note: There is no user-interface equivalent for some of these commands.

See Also

- [Example Programs](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

CSET:CATalog?

This command replaces [SENS:CORR:CSET:CAT?](#)

(Read-only) Returns the names of Cal Sets stored on the PNA.

Parameters None

Examples

```
CSET:CAT?
```

Returns:

```
"CalSet_0913,CalSet_1,CalSet_2,CalSet_3,CalSet_4,CH1_CALREG,CH31_CALREG,MyCalAll_SMC_002,MyCalAll_STD_001"
```

Return Type Comma-separated string of names

Default Not Applicable

CSET:DALL

(Write-only) Deletes ALL Cal Sets from the PNA, including phase reference and Global Delta Match Cal Sets.

Parameters None

Examples

```
CSET:DALL
```

Query Syntax Not Applicable

Default Not Applicable

CSET:DATE? <string>

(Read-only) Returns the (year, month, day) that the specified Cal Set was last saved.

See Also

[MMEM:DATE?](#)

[MMEM:TIME?](#)

[CSET:TIME?](#)

Parameters

<string> Cal Set name.

Examples

```
CSET:DATE? "CalSet_11"
```

```
'Returns:
```

```
+2013,+5,+1
```

Return Type Comma-separated integers.

Default Not Applicable

CSET:DEL <string>

This command replaces [SENS:CORR:CSET:DELeTe](#)

(Write-only) Deletes the specified Cal Set from the PNA.

- If the Cal Set is currently being used by a channel, the Cal Set is deleted and correction for the channel is turned off.
- If the Cal Set is not found, no error is returned.

Parameters

<string> Name of the Cal Set to delete. Not case-sensitive.

Examples

```
CSET:DEL "MyCalSet"
```

Query Syntax Not Applicable

Default Not Applicable

CSET:EXISTS? <string>

(Read-only) Returns whether or not the specified Cal Set exists on the PNA.

Parameters

<string> Name or GUID of the Cal Set enclosed in quotes.

The GUID must also be enclosed in curly brackets.

Examples

```
dim check
```

```
check = CSET:EXISTS? "MyCalSet"
```

```
check = CSET:EXISTS? "{7C4EEA5E-40D2-4D70-A048-33BFFE704163}"
```

Return Type Boolean

ON or **1** - Cal Set exists.

OFF or **0** - Cal Set does NOT exist.

Default Not Applicable

CSET:FIXTure:DEEMbed <cs1>,<cs2>,<s2p>,<port>, <compPwr>[,extrap]

(Write-only) De-embeds a fixture from an existing Cal Set based on an S2P file. A new Cal Set is created with the effects of the fixture removed.

When the new Cal Set is applied to a channel, the effects of fixturing are removed from the measurement data. Do NOT enable fixturing. The effects of the fixture are removed when the new Cal Set is selected and correction is turned ON.

Parameters

<cs1> (String) Name of an existing Cal Set which resides on the PNA.

<cs2> (String) Name of new Cal Set which contains updated error terms with fixture de-embedded.

<s2p> (String) Name of the S2P file which characterizes the adapter/fixture.

<port> (Numeric) Port number from which fixture will be de-embedded.

<compPwr> (Boolean)

ON (1) - When the Cal Set contains a power correction array for the fixture port, that array will be compensated for the fixture loss.

Warning: enabling power compensation can result in an increase in test port power and consequently, increased power to the DUT. Use with caution.

OFF (0) - Do not compensate for loss in source power through the fixture.

[extrap] (Boolean) Optional argument.

ON (1) -Applies a simple extrapolation when the S2P file has a narrower frequency range than the Cal Set. The values for the first and last data points are extended in either direction to cover the frequency range of the Cal Set.

OFF (0) - Extrapolation is NOT performed (default setting).

Examples

```
CSET:FIXT:DEEM "MyCalSet","MyNewCalSet","Fixture.s2p",1,1
cset:fixture:deembed
```

```
"MyCalSet", "MyNewCalSet", "Fixture.s2p", 1, 1, 1 'extrapolation is performed if the s2p frequency range is narrower than that of the Cal Set.
```

Query Syntax Not Applicable

Default Not Applicable

CSET:FIXTure:EMBed <cs1>,<cs2>,<s2p>,<port>, <compPwr>[,<extrap>]

(Write-only) Embeds a fixture (usually a matching network) into an existing Cal Set based on an S2P file. A new Cal Set is created with the effects of the matching network included in the correction data.

When the new Cal Set is applied to a channel, the effects of the fixture are included in the measurement data. Do NOT enable fixturing. The effects of the matching network are included when the new Cal Set is selected and correction is turned ON.

Parameters

<cs1> (String) Name of an existing Cal Set which resides on the PNA.

<cs2> (String) Name of new Cal Set which contains updated error terms with fixture embedded.

<s2p> (String) Name of the S2P file which characterizes the fixture / matching network.

<port> (Numeric) Port number to which fixture will be added.

<compPwr> (Boolean)

ON (1) - Increase the source power to compensate for the loss through the fixture. The result is that the specified power level will be correct at the DUT input.

Warning: enabling power compensation can result in an increase in test port power and consequently, increased power to the DUT. Use with caution.

OFF (0) - Do not compensate for loss in source power through the matching network.

[<extrap>] (Boolean) Optional argument.

ON (1) -Applies a simple extrapolation when the S2P file has a narrower frequency range than the Cal Set. The values for the first and last data points are extended in either direction to cover the frequency range of the Cal Set.

OFF (0) - Extrapolation is NOT performed (default setting).

Examples

```
CSET:FIXT:EMB "MyCalSet","MyNewCalSet","Fixture.s2p",1,1  
  
cset:fixture:embed "MyCalSet","MyNewCalSet","Fixture.s2p",1,1,1  
'extrapolation is performed if the s2p frequency range is  
narrower than that of the Cal Set.'
```

Query Syntax Not Applicable

Default Not Applicable

CSET:TIME? <string>

(Read-only) Returns the (hour, minute, second) that the specified Cal Set was last saved. The time is returned in local time as setup in the PNA operating system.

See Also

[CSET:DATE?](#)

[MMEM:DATE?](#)

[MMEM:TIME?](#)

Parameters

<string> Cal Set name.

Examples

```
CSET:TIME? "CalSet_11"  
  
'Returns:  
  
+13,+6,+1'
```

Return Type Comma-separated integers.

Default Not Applicable

Last Modified:

13-May-2013	Added date and time commands
30-Apr-2013	Add curly brackets to CSET:EXIS?
4-Feb-2013	Added CAT, DEL, DALL
6-Dec-2011	Fixed example
30-Nov-2010	Added extrapolation argument
23-Aug-2010	Fixed FIXT
13-Apr-2010	MX New topic

Display Commands

Controls the settings of the front panel screen.

DISPlay:

ANNOtation

| FREQuency[:STATE]

| MESSAge:STATe

| [:STATus]

ARRange

CATalog?

COLor More Commands

ENABle

FSIGN

TMAX

TILE

TOOLbar

| ENTRy[:STATe]

| EXTensions

| KEYS

| MARKer[:STATe]

| MEAS[:STATe]

| STIMulus[:STATe]

| SWEEp[:STATe]

| TRANsform

VISible

WINDow

| ANNOtation

| MARKer

| NUMBer

| RESolution

| RESPonse

| STIMulus

| SINGle[:STATe]

| SIZE

| STATe

| SYMBol

| XPOSITION

| YPOSITION

| TRACe[:STATe]

| CATalog?

| ENABle

| SIZE

```
| [STATe]
| TABLE
| TITLe
  | DATA
  | [STATe]
| TRACe
  | DELeTe
  | FEED
  | GRATicule:GRID:LTYPe
  | MEMory[:STATe]
  | MOVE
  | SElect
  | [STATe]
  | TITLe
    | DATA
    | [:STATe]
  | Y[:SCALe]
    | AUTO
    | COUPle
      | METHod
      | [STATe]
    | PDIVision
    | RLEVel
    | RPOStion
  | Y:AUTO
```

Click on a [blue](#) keyword to view the command details.

[Red](#) keywords are [superseded](#).

See Also

- [Referring to Traces Channels Windows and Meas Using SCPI](#)
- See an [example](#) using some of these commands
- [Synchronizing the PNA and Controller](#)
- [Learn about Screen Setup](#)
- [SCPI Command Tree](#)

DISPlay:ANNOtation:FREQUency[:STATe] <ON | OFF>

(Read-Write) Turns frequency information on the display title bar ON or OFF for all windows.

Parameters

<ON | OFF> **ON** (or 1) - turns frequency annotation ON.
OFF (or 0) - turns frequency annotation OFF.

Examples

```
DISP:ANN:FREQ ON  
display:annotation:frequency:state off
```

Query Syntax DISPlay:ANNotation:FREQuency[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON (1)

DISPlay:ANNotation:MESSage:STATe <ON | OFF>

(Read-Write) Enables and disables error pop-up messages on the display.

Parameters

<ON | OFF> **ON** (or 1) - enables error pop-up messages
OFF (or 0) - disables error pop-up messages

Examples

```
DISP:ANN:MESS:STAT ON  
display:annotation:message:state off
```

Query Syntax DISPlay:ANNotation:MESSage:STATe?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON (1)

DISPlay:ANNotation[:STATus] <ON | OFF>

(Read-Write) Turns the status bar at the bottom of the screen ON or OFF. The status bar displays information for the active window.

Parameters

<ON | OFF> **ON** (or 1) - turns status bar ON.
OFF (or 0) - turns status bar OFF.

Examples

```
DISP:ANN ON  
display:annotation:status off
```

Query Syntax DISPlay:ANNotation[:STATus]?

Return Type Boolean (1 = ON, 0 = OFF)

Default Last state that was set

DISPlay:ARRange <char>

(Write-only) Places EXISTING measurements into pre-configured window arrangements. Overlay, Stack(2), Split(3), and Quad(4) creates new windows. To learn more, see [Arrange Existing Measurements](#).

Parameters

<char> Window arrangement. Choose from:

- TILE - tiles existing windows
- CASCade - overlaps existing windows
- OVERlay - all traces placed in 1 window
- STACk - 2 windows
- SPLit - 3 windows
- QUAD - 4 windows

Examples

```
DISP:ARR CASC  
display:arrange cascade
```

Query Syntax Not Applicable

Default TILE

DISPlay:CATalog?

(Read-only) Returns the existing Window numbers.

To read the window number of the selected trace, use [Calc:Par:WNUM](#).

Return Type String of Character values, separated by commas

Example Two windows with numbers 1 and 2 returns:
"1,2"

Default Not applicable

DISPlay:ENABLE <ON | OFF>

(Read-Write) Specifies whether to disable or enable all analyzer display information **in all windows** in the analyzer application. Marker data is not updated. More CPU time is spent making measurements instead of updating the display.

Parameters

<ON | OFF> **ON** (or 1) - turns the display ON.
OFF (or 0) - turns the display OFF.

Examples

```
DISP:ENAB ON  
display:enable off
```

Query Syntax DISPlay:ENABle?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

DISPlay:FSIGN <ON | OFF>

(Read-Write) Shows or hides the window which displays global pass/fail results.

Parameters

<ON | OFF> **ON** (or 1) - displays the pass/fail dialog
OFF (or 0) - hides the pass/fail dialog

Examples

```
DISP:FSIG ON  
display:fsign off
```

Query Syntax DISPlay:FSIGN?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

DISPlay:TMAX <bool>

(Read-Write) Maximizes (isolates) or restores the active trace in the active window. When turned ON, the active trace is the ONLY trace on the display. All other traces are hidden. [Learn more.](#)

Parameters

<bool> **ON** (or 1) - Maximize / isolates the active trace.

OFF (or 0) - Restores other traces to the normal window setting.

Examples

```
DISP:TMAX ON
display:tmax 0
```

Query Syntax DISPlay:TMAX?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

DISPlay[:TILE] - Superseded

This command is replaced by [DISP:ARRange](#)

(Write-only) Tiles the windows on the screen.

Examples

```
DISP
display:tile
```

Default Not Applicable

DISPlay:TOOLbar:ENTRy[:STATe] <bool>

(Read-Write) Specifies whether to show or hide the active entry toolbar. [See this toolbar.](#)

Parameters

<bool> **ON** (or 1) - Toolbar ON.

OFF (or 0) - Toolbar OFF.

Examples

```
DISP:TOOL:ENTR ON
display:toolbar:entry:state off
```

Query Syntax DISPlay:TOOLbar:ENTRy[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

DISPlay:TOOLbar:EXTensions[:STATe] <bool>

(Read-Write) Specifies whether to show or hide the port extensions toolbar. [See this toolbar.](#)

Parameters

<bool> **ON** (or 1) - Toolbar ON.

OFF (or 0) - Toolbar OFF.

Examples

```
DISP:TOOL:EXT ON  
display:toolbar:extensions:state off
```

Query Syntax DISPlay:TOOLbar:EXTensions[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

DISPlay:TOOLbar:KEYS[:STATe] <bool>

(Read-Write) Specifies whether to show or hide the virtual hardkeys on the PNA display. These are primarily used when the PNA is accessed remotely using [VNC](#) or Windows Remote Desktop.

Parameters

<bool> **ON** (or 1) - Keys ON.

OFF (or 0) - Keys OFF.

Examples

```
DISP:TOOL:KEYS ON  
display:toolbar:keys:state off
```

Query Syntax DISPlay:TOOLbar:KEYS [:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

DISPlay:TOOLbar:MARKer[:STATe] <bool>

(Read-Write) Specifies whether to show or hide the marker toolbar. [See this toolbar.](#)

Parameters

<bool> **ON** (or 1) - Toolbar ON.
OFF (or 0) - Toolbar OFF.

Examples

```
DISP:TOOL:MARK ON  
display:toolbar:marker:state off
```

Query Syntax DISPlay:TOOLbar:MARKer[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

DISPlay:TOOLbar:MEASurement[:STATe] <bool> OBSOLETE

This toolbar was eliminated with A.10.00

(Read-Write) Specifies whether to show or hide the measurement toolbar.

Parameters

<bool> **ON** (or 1) - Toolbar ON.
OFF (or 0) - Toolbar OFF.

Examples

```
DISP:TOOL:MEAS ON  
display:toolbar:measurement:state off
```

Query Syntax DISPlay:TOOLbar:MEASurement[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

DISPlay:TOOLbar:STIMulus[:STATe] <bool> OBSOLETE

This toolbar was eliminated with A.10.00

(Read-Write) Specifies whether to show or hide the stimulus toolbar.

Parameters

<bool> **ON** (or 1) - Toolbar ON.

OFF (or 0) - Toolbar OFF.

Examples

```
DISP:TOOL:STIM ON
```

```
display:toolbar:stimulus:state off
```

Query Syntax DISPlay:TOOLbar:STIMulus[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

DISPlay:TOOLbar:SWEep[:STATe] <bool> OBSOLETE

This toolbar was eliminated with A.10.00

(Read-Write) Specifies whether to show or hide the sweep control toolbar.

Parameters

<bool> **ON** (or 1) - Toolbar ON.

OFF (or 0) - Toolbar OFF.

Examples

```
DISP:TOOL:SWE ON
```

```
display:toolbar:sweep:state off
```

Query Syntax DISPlay:TOOLbar:SWEep[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

DISPlay:TOOLbar:TRANSform[:STATe] <bool>

(Read-Write) Specifies whether to show or hide the port extensions toolbar. [See this toolbar.](#)

Parameters

<bool> **ON** (or 1) - Toolbar ON.
OFF (or 0) - Toolbar OFF.

Examples

```
DISP:TOOL:TRAN ON  
display:toolbar:transform:state off
```

Query Syntax DISPlay:TOOLbar:TRANsform[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

DISPlay:VISible <ON | OFF>

(Read-Write) Makes the PNA application visible or not visible. In the Not Visible state, the analyzer cycle time for making measurements, and especially data transfer, can be significantly faster because the display does not process data.

Parameters

<ON | OFF> **ON** (or 1) - PNA app is visible
OFF (or 0) - PNA app is NOT visible

Examples

```
DISP:VIS ON  
display:visible off
```

Query Syntax DISPlay:VISible?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

DISPlay:WINDow<wnum>:ANNotation:MARKer:NUMBer <num>

This command replaces [DISP:WIND:ANN:MARK:SINGLE](#)

(Read-Write) Sets the number of marker readouts to display per trace. Display up to 20 marker readouts per window.

See other SCPI [Marker](#) commands. Learn more about [Marker readout](#).

Parameters

- <wnum> Any existing window number. If unspecified, value is set to 1.
- <num> Number of marker readouts to display. Choose a value between 1 and 10.

Examples

```
DISP:WIND:ANN:MARK:NUMB 7  
display:window:annotation:marker:number 2
```

Query Syntax DISPlay:WINDow:ANNotation:MARKer:NUMBer?

Return Type Numeric

Default 5

DISPlay:WINDow<wnum>:ANNotation:MARKer:RESolution:STIMulus <num>

(Read-Write) For the X-axis (stimulus), sets the number digits to display after the decimal point in marker readouts.

See other SCPI [Marker](#) commands. Learn more about [Marker readout](#).

Parameters

- <wnum> Any existing window number. If unspecified, value is set to 1.
- <num> Number of digits to display. Choose a value between 2 and 6.

Examples

```
DISP:WIND:ANN:MARK:RES:STIM 2  
display:window:annotation:marker:resolution:stimulus 4
```

Query Syntax DISPlay:WINDow:ANNotation:MARKer:RESolution:STIMulus?

Return Type Numeric

Default 3

DISPlay:WINDow<wnum>:ANNotation:MARKer:RESolution:RESPonse <num>

(Read-Write) For the Y-axis (response), sets the number digits to display after the decimal point in marker readouts.

See other SCPI [Marker](#) commands. Learn more about [Marker readout](#).

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1.

<num> Number of digits to display. Choose a value between 1 and 4.

Examples

```
DISP:WIND:ANN:MARK:RES:RESP 1  
display:window:annotation:marker:resolution:stimulus 2
```

Query Syntax DISPlay:WINDow:ANNotation:MARKer:RESolution:RESPonse?

Return Type Numeric

Default 2

DISPlay:WINDow<wnum>:ANNotation:MARKer:SINGLE[:STATe] <bool> - Superseded

Note: This command is replaced by [DISP:WIND:ANN:MARK:NUMB](#)

(Read-Write) Either shows marker readout of only the active trace or other traces simultaneously.

See other SCPI [Marker](#) commands. Learn more about [Marker readout](#).

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1.

<bool> **ON** (or 1) - Shows the readout of only the active marker for each trace.

OFF (or 0) - Shows up to 5 marker readouts per trace, up to 20 total readouts.

Examples

```
DISP:WIND:ANN:MARK:SING ON  
display:window:annotation:marker:single off
```

Query Syntax DISPlay:WINDow:ANNotation:MARKer:SINGLE?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

DISPlay:WINDow<wnum>:ANNotation:MARKer:SIZE <char>

(Read-Write) Specifies the size of the marker readout text. See other SCPI [Marker](#) commands. Learn more about [Marker readout](#).

Parameters

- <wnum> Any existing window number. If unspecified, value is set to 1.
- <char> Readout text size. Choose from: **NORMAL** | **LARGE**

Examples

```
DISP:WIND:ANN:MARK:SIZE LARG
display:window:annotation:marker:size normal
```

Query Syntax DISPlay:WINDow:ANNotation:MARKer:SIZE?

Return Type Character

Default NORMAL

DISPlay:WINDow<wnum>:ANNotation:MARKer[:STATe] <ON | OFF>

(Read-Write) Specifies whether to show or hide the Marker readout (when markers are ON) on the selected window. See other SCPI [Marker](#) commands. Learn more about [Marker readout](#).

Parameters

- <wnum> Any existing window number. If unspecified, value is set to 1.
- <ON | OFF> **ON** (or 1) - turns marker readout ON.
OFF (or 0) - turns marker readout OFF.

Examples

```
DISP:WIND:ANN:MARK ON
display:window:annotation:marker:state off
```

Query Syntax DISPlay:WINDow:ANNotation:MARKer[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

DISPlay:WINDow<wnum>:ANNotation:MARKer:SYMBOL <char>

(Read-Write) Sets the symbol to display for marker position.

See other SCPI [Marker](#) commands.

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1.

<char> Marker symbol. Choose from:

TRIangle

FLAG

LINE

[See pictures of each](#)

Examples

```
DISP:WIND:ANN:MARK:SYMB TRI
```

```
display:window:annotation:marker:symbol line
```

Query Syntax DISPlay:WINDow:ANNotation:MARKer:SYMBol?

Return Type Character

Default TRIangle

DISPlay:WINDow<wnum>:ANNotation:MARKer:XPOSition <num>

(Read-Write) Sets the X-axis position of marker readouts. Readouts are right-justified at the specified position.

See other SCPI [Marker](#) commands. Learn more about [Marker readout](#).

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1.

<num> X-axis position. Choose a value between 1 (far left) and 10 (far right).

Examples

```
DISP:WIND:ANN:MARK:XPOS 1.5
```

```
display:window:annotation:marker:xposition 5
```

Query Syntax DISPlay:WINDow:ANNotation:MARKer:XPOSition?

Return Type Numeric

Default 10

DISPlay:WINDow<wnum>:ANNotation:MARKer:YPOSition <num>

(Read-Write) Sets the Y-axis position of marker readouts. Readouts are top-justified at the specified position.

See other SCPI [Marker](#) commands. Learn more about [Marker readout](#).

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1.

<num> Y-axis position. Choose a value between 1 (bottom) and 10 (top).

Examples

```
DISP:WIND:ANN:MARK:YPOS 1.5  
display>window:annotation:marker:yposition 5
```

Query Syntax DISPlay:WINDow:ANNotation:MARKer:YPOSition?

Return Type Numeric

Default 10

DISPlay:WINDow<wnum>:ANNotation[:TRACe][:STATe] <ON | OFF>

(Read-Write) Specifies whether to show or hide the Trace Status buttons on the left of the display.

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1.

<ON | OFF> **ON** (or 1) - turns the buttons ON.
OFF (or 0) - turns the buttons OFF.

Examples

```
DISP:WIND:ANN ON  
display>window:annotation:trace:state off
```

Query Syntax DISPlay:WINDow:ANNotation[:TRACe][:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

DISPlay:WINDow<wnum>:CATalog?

(Read-only) Returns the trace numbers for the specified window.

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1.

Example

```
Window 1 with four traces:  
DISPlay:WINDow1:CATalog?  
Returns:  
"1,2,3,4"
```

Return Type String of Character values separated by commas

Default Not applicable

DISPlay:WINDow<wnum>:ENABle <ON | OFF>

(Read-Write) Specifies whether to disable or enable all analyzer display information **in the specified window**. Marker data is not updated. More CPU time is spent making measurements instead of updating the display.

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1.

<ON | OFF> **ON** (or 1) - turns the display ON.
OFF (or 0) - turns the display OFF.

Examples

```
DISP:WIND:ENABle ON  
display:window1:enable off
```

Query Syntax DISPlay:WINDow<wnum>:ENABle?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

DISPlay:WINDow<wnum>:SIZE <char>

(Read-Write) Sets or returns the window setting of Maximized, Minimized, or Normal. To arrange all of the windows, use [DISP:ARR](#).

Parameters

- <wnum> Any existing window number. If unspecified, value is set to 1
- <char> Window size. Choose from:

MIN | MAX | NORM

Examples `DISP:WIND:SIZE MAX`
`display:window:size norm`

Query Syntax `DISPlay:WINDow:SIZE?`

Default Not Applicable

DISPlay:WINDow<wnum>[:STATe] <ON | OFF>

(Read-Write) **Write** to create or delete a window on the screen or **Read** whether a window is present.

Parameters

- <wnum> Window number to create; choose any integer between **1** and the [maximum number of windows allowed in the PNA](#).
- <ON | OFF> **ON** (or 1) - The window <wnum> is created.
OFF (or 0) - The window <wnum> is deleted.

Examples `DISP:WIND ON`
`display:window2:state off`

Query Syntax `DISPlay:WINDow<wnum>[:STATe]?`

Return Type Boolean (1 = ON, 0 = OFF)

Default Window number "1" **ON**

DISPlay:WINDow<wnum>:TABLE <char>

(Read-Write) **Write** to show the specified table at the bottom of the analyzer screen or **Read** to determine what table is visible.

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1

<char> Table to show. Choose from:
OFF | MARKer | LIMit | SEGMENT

Examples `DISP:WIND:TABLE SEGMENT
display:window:table off`

Query Syntax `DISPlay:WINDow:TABLE?`

Default OFF

DISPlay:WINDow<wnum>:TITLe:DATA <string>

(Read-Write) Sets data in the window title area. The title is turned ON and OFF with [DISP:WIND:TITL:STATOFF](#).

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1.

<string> Title to be displayed. Any characters, enclosed with quotes. If the title string exceeds 50 characters, an error will be generated and the title not accepted. Newer entries replace (not append) older entries.

Examples `DISP:WIND:TITL:DATA 'hello'
display:window2:title:data 'hello'`

Query Syntax `DISPlay:WINDow<wnum>:TITLe:DATA?`

Return Type String

Default NA

DISPlay:WINDow<wnum>:TITLe[:STATe] <ON | OFF>

(Read-Write) Turns display of the title string ON or OFF. When OFF, the string remains, ready to be redisplayed when turned back ON.

Parameters

- <wnum> Any existing window number. If unspecified, value is set to 1
- <ON | OFF> **ON** (or 1) - turns the title string ON.
OFF (or 0) - turns the title string OFF.

Examples

```
DISP:WIND:TITL ON  
Display:window1:title:state off
```

Query Syntax DISPlay:WINDow<wnum>:TITLe[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

DISPlay:WINDow<wnum>:TRACe<tnum>:DELete

(Write-only) Deletes the specified trace from the specified window. The measurement parameter associated with the trace is not deleted.

Parameters

- <wnum> Any existing window number. If unspecified, value is set to 1.
- <tnum> The number of the trace to be deleted; if unspecified, value is set to 1.

Note: This is **NOT** the trace number of the channel which appears as the [Tr annotation](#) on the Trace Status display. This <tnum> is the trace number within the specified window, and is used **ONLY** for remote programs.

Examples

```
DISP:WIND:TRAC:DEL  
display:window2:trace2:delete
```

Query Syntax Not Applicable

Default Not Applicable

DISPlay:WINDow<wnum>:TRACe<tnum>:FEED <name>

(Write-only) Creates a new trace <tnum> and associates (feeds) a measurement <name> to the specified window<wnum>. This command should be executed immediately after creating a new measurement with [CALC:PAR:DEF<name>,<parameter>](#).

To feed the same measurement to multiple traces, create another measurement with the same <parameter>, but different <name>, using the CALC:PAR:DEF command. The analyzer will collect the data only once.

Parameters

- <wnum> Any existing window number. If unspecified, value is set to 1.
- <tnum> Trace number to be created. Choose any Integer between 1 and the PNA [maximum number of traces per window](#) allowed.

Note: This is **NOT** the trace number of the channel which appears as the [Tr annotation](#) on the Trace Status display. This <tnum> is the trace number within the specified window, and is used **ONLY** for remote programs.

- <name> Name of the measurement that was defined with
CALC:PAR:DEF<name>,<parameter>

Examples

```
DISP:WIND:TRAC:FEED 'test '  
display>window2:trace2:feed 'test '
```

Query Syntax Not applicable

Default "CH1_S11"

DISPlay:WINDow:TRACe:GRATicule:GRID:LTYPE <value>

(Read-Write) Sets and returns the grid line type (solid | dotted) for all open windows. Grid is returned to solid when the PNA is Preset. [Learn more.](#)

Parameters

- <value> Line type. Choose from:

SOLid - solid lines

DOTTed - dotted lines

Examples

```
DISP:WIND:TRAC:GREAT:GRID:LTYPE SOL  
display>window:trace:graticule:grid:ltype dotted
```

Query Syntax DISPlay:WINDow:TRACe:GRATicule:GRID:LTYPE?

Return Type Character

Default SOLID

DISPlay:WINDow<wnum>:TRACe<tnum>:MEMory[:STATe] <ON | OFF>

(Read-Write) Turns the memory trace ON or OFF.

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1.

<tnum> Any existing trace number; if unspecified, value is set to 1.

Note: This is **NOT** the trace number of the channel which appears as the [Tr annotation](#) on the Trace Status display. This <tnum> is the trace number within the specified window, and is used **ONLY** for remote programs.

<ON | OFF> **ON** (or 1) - turns the memory trace ON.
OFF (or 0) - turns the memory trace OFF.

Examples

```
DISP:WIND:TRAC:MEM ON  
display:window2:trace2:memory:state off
```

Query Syntax DISPlay:WIND<wnum>:TRACe<tnum>:MEMory[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

DISPlay:WINDow<fromWin>:TRACe<tnum>:MOVE <toWin>

(Write-only) Moves a trace from one window to another window.

Parameters

<fromWin> Window number to move the trace **from**. If unspecified, value is set to 1.

Use [Disp:Cat?](#) to read the existing window numbers.

<tnum> Trace number to be moved. If unspecified, value is set to 1.

Use [Disp:Wind:Cat?](#) to read the trace numbers in an existing window.

Note: This is **NOT** the trace number of the channel which appears as the [Tr annotation](#) on the Trace Status display. This <tnum> is the trace number within the specified window, and is used **ONLY** for remote programs.

<toWin> Number of the window to move the trace **to**. If the window does not exist, it will be created.

Examples

```
DISP:WIND:TRAC2:MOVE 2  
display:window2:trace2:move 1
```

Query Syntax Not applicable

Default Not applicable

DISPlay:WINDow<wnum>:TRACe<tnum>:SElect

(Write-only) Activates the specified trace in the specified window for front panel use.

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1.

Use [Disp:Cat?](#) to read the existing window numbers.

<tnum> Any existing trace number; if unspecified, value is set to 1.

Note: This is **NOT** the trace number of the channel which appears as the [Tr annotation](#) on the Trace Status display. This <tnum> is the trace number within the specified window, and is used **ONLY** for remote programs.

Examples

```
DISP:WIND:TRAC:SEL
display:window2:trace2:select
```

Query Syntax Not applicable

Default NA

DISPlay:WINDow<wnum>:TRACe<tnum>[:STATe] <ON | OFF>

(Read-Write) Turns the display of the specified trace in the specified window ON or OFF. When OFF, the measurement behind the trace is still active.

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1.

Use [Disp:Cat?](#) to read the existing window numbers.

<tnum> Any existing trace number; if unspecified, value is set to 1.

Use [Disp:Wind:Cat?](#) to read the trace numbers in an existing window.

Note: This is **NOT** the trace number of the channel which appears as the [Tr annotation](#) on the Trace Status display. This <tnum> is the trace number within the specified window, and is used **ONLY** for remote programs.

<ON | OFF> **ON** (or 1) - turns the trace ON.
OFF (or 0) - turns the trace OFF.

Examples

```
DISP:WIND:TRAC ON
display:window2:trace2:state off
```

Query Syntax DISPlay:WIND<wnum>:TRACe<tnum>[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

DISPlay:WINDow<wnum>:TRACe<tnum>:TITLe:DATA <string>

(Read-Write) Writes and read data to the trace title area. The trace title is embedded in the trace status field. [Learn more about Trace Titles.](#)

Newer entries replace (not append) older entries. The title is turned ON and OFF with [DISP:WIND:TRAC:TITL:STAT.](#)

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1.

Use [Disp:Cat?](#) to read the existing window numbers.

<tnum> Trace number of the specified window. If unspecified, value is set to 1. Use [Display:Cat?](#) to read the window numbers. Use [Disp:Window:Cat?](#) to read the trace numbers of the specified window.

Note: This is **NOT** the trace number of the channel which appears as the [Tr annotation](#) on the Trace Status display. This <tnum> is the trace number within the specified window, and is used **ONLY** for remote programs.

<string> Title to be displayed. Any characters (not spaces) enclosed with quotes.

Examples

```
DISP:WIND:TRAC:TITL:DATA 'MyNewMeas'  
display>window2:trace3:title:data 'hello'
```

Query Syntax DISPlay:WINDow<wnum>:TRACe<tnum>TITLe:DATA?

Return Type String

Default Not Applicable

DISPlay:WINDow<wnum>:TRACe<tnum>:TITLe[:STATe] <bool>

(Read-Write) Turns display of the Trace Title ON or OFF. When turned OFF, the previous trace title returns. Set a new trace title using [DISP:WIND:TRAC:TITL:DATA](#)

[Learn more about Trace Titles](#)

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1

Use [Disp:Cat?](#) to read the existing window numbers.

<tnum> Trace number of the specified window. If unspecified, value is set to 1. Use [Display:Cat?](#) to read the window numbers. Use [Disp:Window:Cat?](#) to read the trace numbers of the specified window.

Note: This is **NOT** the trace number of the channel which appears as the [Tr annotation](#) on the Trace Status display. This <tnum> is the trace number within the specified window, and is used **ONLY** for remote programs.

<bool> **ON** (or 1) - turns the title ON.

OFF (or 0) - turns the title OFF.

Examples

```
DISP:WIND:TRAC:TITL ON  
Display:window2:trace3:title:state off
```

Query Syntax DISPlay:WINDow<wnum>:TRACe<tnum>:TITLe[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

DISPlay:WINDow<wnum>:TRACe<tnum>:Y[:SCALe]:AUTO

(Write-only) Performs an **Autoscale** on the specified trace in the specified window, providing the best fit display.

Autoscale is performed only when the command is sent; it does NOT keep the trace autoscaled indefinitely.

Autoscale behaves differently when [scale coupling](#) is enabled. How it behaves depends on the scale coupling method. [Learn more.](#)

See Also, [DISPlay:WINDow:Y:AUTO](#) which performs an Autoscale All.

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1.

Use [Disp:Cat?](#) to read the existing window numbers.

<tnum> Any existing trace number; if unspecified, value is set to 1.

Use [Disp:Wind:Cat?](#) to read the trace numbers in an existing window.

Note: This is **NOT** the trace number of the channel which appears as the [Tr annotation](#) on the Trace Status display. This <tnum> is the trace number within the specified window, and is used **ONLY** for remote programs.

Examples

```
DISP:WIND:TRAC:Y:AUTO
display>window2:trace2:y:scale:auto
```

Query Syntax Not applicable

Default Not applicable

DISPlay:WINDow:TRACe:Y[:SCALE]:COUPlE:METhod <char>

(Read-Write) Sets and returns the method of scale coupling. [Learn more](#) about Scale coupling.

Parameters

<char> **OFF** - NO scale coupling for any windows.

WINDow - Scale settings are coupled for traces in each window.

ALL - Scale settings are coupled for traces in ALL selected windows.

Enable the selected windows using [DISP:WIND:TRAC:Y:COUP ON](#)

Examples

```
DISP:WIND:TRAC:Y:COUP:METH ALL
```

```
Display>window2:trace:y:scale:method window
```

Query Syntax DISPlay:WINDow:TRACe:Y[:SCALe]:COUPle:METhod?

Return Type Character

Default OFF

DISPlay:WINDow<wnum>:TRACe:Y[:SCALe]:COUPle[:STATe] <bool>

(Read-Write) Enables and disables scale coupling for the specified window. [Learn more](#) about Scale coupling.

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1

Use [Disp:Cat?](#) to read the existing window numbers.

<bool> **ON** (or 1) - Scale coupling enabled for specified window.

OFF (or 0) - Scale coupling disabled for specified window.

Examples

```
DISP:WIND:TRAC:Y:COUP ON
```

```
Display>window2:trace:y:scale:couple:state off
```

Query Syntax DISPlay:WINDow<wnum>:TRACe:Y[:SCALe]:COUPle[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

DISPlay:WINDow<wnum>:TRACe<tnum>:Y[:SCALe]:PDIVision <num>

(Read-Write) Sets the Y axis **Per Division** value of the specified trace in the specified window.

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1.

Use [Disp:Cat?](#) to read the existing window numbers.

<tnum> Any existing trace number; if unspecified, value is set to 1.

Use [Disp:Wind:Cat?](#) to read the trace numbers in an existing window.

Note: This is **NOT** the trace number of the channel which appears as the [Tr annotation](#) on the Trace Status display. This <tnum> is the trace number within the specified window, and is used **ONLY** for remote programs.

<num> Units / division value. The range of acceptable values is dependent on format and domain.

Note: This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
DISP:WIND:TRAC:Y:PDIV 1
display>window2:trace2:y:scale:pdivision maximum
```

Query Syntax DISPLAY:WINDow<wnum>:TRACe<tnum>:Y[:SCALe]:PDIVision?

Return Type Numeric

Default 10

DISPlay:WINDow<wnum>:TRACe<tnum>:Y[:SCALe]:RLEVel <num>

(Read-Write) Sets the Y axis Reference Level of the specified trace in the specified window.

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1.

Use [Disp:Cat?](#) to read the existing window numbers.

<tnum> Any existing trace number; if unspecified, value is set to 1.

Use [Disp:Wind:Cat?](#) to read the trace numbers in an existing window.

Note: This is **NOT** the trace number of the channel which appears as the [Tr annotation](#) on the Trace Status display. This <tnum> is the trace number within the specified window, and is used **ONLY** for remote programs.

<num> Reference level value. The range of acceptable values is dependent on format

and domain.

Note: This command will accept MIN or MAX instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
DISP:WIND:TRAC:Y:RLEV 0  
display>window2:trace2:y:scale:rlevel minimum
```

Query Syntax DISPlay:WINDow<wnum>:TRACe<tnum>:Y[:SCALe]:RLEVel?

Return Type Numeric

Default Not Applicable

DISPlay:WINDow<wnum>:TRACe<tnum>:Y[:SCALe]:RPOSition <num>

(Read-Write) Sets the **Reference Position** of the specified trace in the specified window

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1.

Use [Disp:Cat?](#) to read the existing window numbers.

<tnum> Any existing trace number; if unspecified, value is set to 1.

Use [Disp:Wind:Cat?](#) to read the trace numbers in an existing window.

Note: This is **NOT** the trace number of the channel which appears as the [Tr annotation](#) on the Trace Status display. This <tnum> is the trace number within the specified window, and is used **ONLY** for remote programs.

<num> Reference position on the screen measured in horizontal graticules from the bottom. The range of acceptable values is dependent on format and domain.

Note: This command will accept MIN or MAX instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
DISP:WIND:TRAC:Y:RPOS 0  
display>window2:trace2:y:rposition maximum
```

Query Syntax DISPlay:WINDow<wnum>:TRACe<tnum>:Y[:SCALe]:RPOSition?

Return Type Numeric

Default 5

DISPlay:WINDow<wnum>:Y:AUTO

(Write-only) Scales **ALL** of the traces to fit in the same window. This is equivalent to "Autoscale All" from the front panel.

Autoscale behaves differently when [scale coupling](#) is enabled. How it behaves depends on the scale coupling method. [Learn more.](#)

Autoscale is performed only when the command is sent; it does NOT keep the trace autoscaled indefinitely.

See Also, [DISPlay:WINDow:TRACe:Y:AUTO](#) which Autoscales only the specified trace.

Parameters

<wnum> Any existing window number. If unspecified, value is set to 1.

Use [Disp:Cat?](#) to read the existing window numbers.

Examples

```
DISP:WIND:Y:AUTO
display>window2:y:auto
```

Query Syntax Not applicable

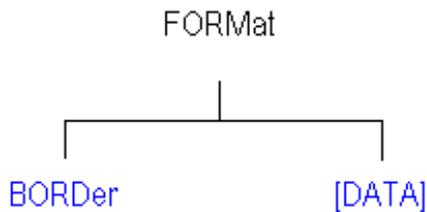
Default Not applicable

Last modified:

29-Oct-2013	Fixed several errors in tree
24-Jun-2013	Removed three toolbars
3-May-2013	Added toolbar state commands
13-Feb-2013	Modified FEED command
16-Mar-2012	Fixed link to max traces (JE)
25-Jul-2011	Added trace move
11-Jan-2011	Minor edits
15-Sep-2010	Added autoscale all and scale coupling note.
6-Aug-2010	Added marker readout (A.09.30)
16-Mar-2010	Added grid line (9.2)
23-Mar-2009	Added <tnum> note
26-Aug-2008	Added TraceMax
12-Sep-2006	Modified for number of windows.

Format Commands

Specifies the way that data will be transferred when moving large amounts of data.



Click on a [blue](#) keyword to view the command details.

See Also

- [Example Programs](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

FORMat:BORDer <char>

(Read-Write) Set the byte order used for GPIB data transfer. Some computers read data from the analyzer in the reverse order. This command is only implemented if FORMAT:DATA is set to :REAL.

If FORMAT:DATA is set to :ASCII, the swapped command is ignored.

Parameters

<char> Choose from:

NORMal - Use when your controller is anything other than an IBM compatible computers.

SWAPped - for IBM compatible computers.

Note: Use **NORMal** if you are using VEE, LabView, or T&M Tool kit.

Examples

```
FORM:BORD SWAP  
format:border normal
```

Query Syntax FORMat:BORDer?

Return Type Character

Default Normal

FORMat[:DATA] <char>

(Read-Write) Sets the data format for transferring measurement data and frequency data.

- To transfer measurement data, use [CALC:DATA](#).
- To transfer Cal Set data, use [SENS:CORR:CSET:DATA](#)
- To transfer Source Power correction data, use:
 - [SOURce:POWer:CORRection:COLLect:TABLE:DATA](#)
 - [SOURce:POWer:CORRection:COLLect:TABLE:FREQuency](#)
 - [SOURce:POWer:CORRection:DATA](#)
- To transfer FIFO buffer data, use [SYST:FIFO:DATA?](#)

The following commands transfer frequency data. Use <REAL, 64>

- [CALC:DATA:SNP?](#)
- [CALC:DATA:SNP:PORTS?](#)
- [CALC:X?](#)
- [SENS:X?](#)

Use [FORMat:BORDER](#) to change the byte order. Use “NORMal” when transferring a binary block from LabView or Vee. For other programming languages, you may need to SWAP the byte order.

Parameters

<char> In the PNA, measurement data is stored as 32 bit and frequencies stored as 64 bit. Therefore, use REAL,32 when getting data and REAL,64 when getting frequencies. That way you are guaranteed to avoid losing any precision as well as getting the maximum speed on the data transfer.

Choose from:

- **REAL,32** - (default value for REAL) Best for transferring large amounts of measurement data. Can cause rounding errors in frequency data.
- **REAL,64** - Slower but has more significant digits than REAL,32. REQUIRED to accurately represent frequency data. See above list for commands which transfer frequency information.

- **ASCii,0** - The easiest to implement, but very slow. Use when you have small amounts of data to transfer.

Note The REAL,32 and REAL,64 arguments transfer data in block format as explained in [Transferring Measurement Data](#).

Examples

```
FORM REAL,64  
format:data ascii
```

Query Syntax FORMat:DATA?

Return Type Character,Character

Default ASCii,0

Syst:Preset does NOT reset this command.

However, *RST does reset this command to ASCii,0

Last Modified:

10-Sep-2012	Added 64 for frequency
6-Sep-2012	Added Form:BOrder link and note (JE)
18-May-2009	Added Real 64 note
17-Sep-2008	Added *RST vs Syst:Pres note.

Hardcopy Command

Controls printing of the PNA screen and optional data to a printer or a file.

HCOPY:

[DPRinter](#)

[FILE](#)

[\[IMMediate\]](#)

ITEM

| [AWINdow](#)

| [CTABLE](#)

| [GPFail](#)

| [LOGO](#)

| [MKRData](#)

| [PNUMber](#)

| [SEGData](#)

| [SWINdow](#)

| [TIME](#)

| [TTABLE](#)

| [WFRaction](#)

| [WINDowns](#)

PAGE

| [DIMension](#)

 | [LLEFt](#)

 | [URIGHt](#)

| [ORientation](#)

| [SIZE](#)

SDUMP

| [DATA?](#)

 | [FORMat](#)

[PRINters?](#)

Click on a [blue](#) keyword to view the command details.

[Red](#) commands are superseded or obsolete.

See Also

- [Learn more about PNA Printing](#)
- [Example Programs](#)

- [Synchronizing the PNA and Controller](#)
-

HCOPY:DPRinter <string>

(Read-Write) Sets the default printer and selects as the current printer. Use [HCOPY:PRINters?](#) to return a list of locally installed printers.

This setting survives instrument preset and PNA application restart.

Parameters

<string> Name of the printer to become the default.

Examples

```
HCOPY:DPR "MyPrinter"  
hcopy:dprinter "YourPrinter"
```

Query Syntax HCOPY:DPRinter?

Return Type String

Default Not Applicable

HCOPY:FILE <filename>

(Write-only) Saves the screen image to a file. The image does NOT include the optional print data invoked by many HCOPY commands.

Parameters

<filename> Name of the file to save the screen to. The file is saved to the current working directory unless a valid full path name is specified.

Use one of the following suffixes:

.bmp - not recommended due to large file size

.jpg - not recommended due to poor quality

.png - recommended

Examples

```
HCOPY:FILE "myFile.png"  
hcopy:file "c:/data/myfile.png"
```

Query Syntax Not Applicable

Default Not Applicable

HCOPy[:IMMEDIATE]

(Write-only) Prints the screen to the default printer.

Examples `HCOP
hcopy:immediate`

Query Syntax Not applicable

Default Not Applicable

HCOPy:ITEM:AWINDOW[:STATE] <bool>

(Read-Write) When ON, prints only the Active window. When OFF, prints all windows.

This setting survives instrument preset and PNA application restart.

Parameters

<bool> Active window state. Chose from:

OFF or (0) - Print ALL windows.

ON or (1) - Print Active window only.

Examples `HCOP:ITEM:AWIN 1
hcopy:item:awindow:state off`

Query Syntax HCOPy:ITEM:AWINDOW[:STATE]?

Return Type Boolean

Default OFF (0)

HCOPy:ITEM:CTABLE[:STATE] <bool>

(Read-Write) When ON, prints the channel settings table.

This setting survives instrument preset and PNA application restart.

Parameters

<bool> Channel table print state. Chose from:

OFF or (0) - Does NOT print the channel settings table.

ON or (1) - Prints channel settings table.

Examples

```
HCOPY:ITEM:CTAB 1
hcopy:item:ctable:state off
```

Query Syntax HCOpy:ITEM:CTABle[:STATe]?

Return Type Boolean

Default OFF (0)

HCOPY:ITEM:GPFail[:STATe] <bool>

(Read-Write) When ON, prints the [Global Pass/Fail](#) status in the page header.

This setting survives instrument preset and PNA application restart.

Parameters

<bool> Pass / Fail print state. Chose from:

OFF or (0) - Does NOT print Pass / Fail status.

ON or (1) - Print Pass / Fail status

Examples

```
HCOPY:ITEM:GPF 1
hcopy:item:gpfail:state off
```

Query Syntax HCOpy:ITEM:GPFail[:STATe]?

Return Type Boolean

Default OFF (0)

HCOPY:ITEM:LOGO[:STATe] <bool>

(Read-Write) When ON, prints the Agilent Technologies logo in the page header.

This setting survives instrument preset and PNA application restart.

Parameters

<bool> Agilent logo print state. Chose from:

OFF or (0) - Prints the Agilent logo.

ON or (1) - Does NOT print the Agilent logo.

Examples

```
HCOPY:ITEM:LOGO 1
```

```
hcopy:item:logo:state off
```

Query Syntax HCOpy:ITEM:LOGO[:STATe]?

Return Type Boolean

Default OFF (0)

HCOPY:ITEM:MKRData[:STATe] <bool>

(Read-Write) When ON, includes marker data as part of the [trace attributes table](#).

To print marker data, [HCOPY:ITEM:TTABLE](#) must also be set to ON.

This setting does not affect the limited [marker readout data](#) that can be displayed in the measurement window.

This setting survives instrument preset and PNA application restart.

Parameters

<bool> Marker data print state. Chose from:

OFF or (0) - Does NOT print Marker data.

ON or (1) - Print Marker data.

Examples

```
HCOPY:ITEM:MKRD 1
```

```
hcopy:item:mkrdata:state off
```

Query Syntax HCOpy:ITEM:MKRData[:STATe]?

Return Type Boolean

Default OFF (0)

HCOPY:ITEM:PNUMBER[:STATE] <bool>

(Read-Write) When ON, prints page numbers (1 of n) in the header at the top of each page.

This setting survives instrument preset and PNA application restart.

Parameters

<bool> Page number print state. Chose from:

OFF or (0) - Does NOT print page numbers.

ON or (1) - Print page numbers.

Examples

```
HCOP:ITEM:PNUM 1
```

```
hcopy:item:pnumber:state off
```

Query Syntax HCOPY:ITEM:PNUMBER[:STATE]?

Return Type Boolean

Default OFF (0)

HCOPY:ITEM:SEGDATA[:STATE] <bool> - **Obsolete**

Note: This command no longer works beginning with A.09.40

(Read-Write) When ON, includes ALL segment data as part of the [channel settings table](#).

To print ALL segment data, [HCOP:ITEM:CTAB](#) must also be set to ON.

This setting survives instrument preset and PNA application restart.

Parameters

<bool> Expanded segment data print state. Chose from:

OFF or (0) - Does NOT print expanded segment data, but summary data is printed.

ON or (1) - Print expanded segment data.

Examples

```
HCOP:ITEM:SEGD 1
```

```
hcopy:item:segdata:state off
```

Query Syntax HCOPY:ITEM:SEGDATA[:STATE]?

Return Type Boolean

Default OFF (0)

HCOPY:ITEM:SWINDOW[:STATe] <bool>

(Read-Write) When ON, prints a single measurement window per page. When OFF, prints up to four measurement windows per page.

This setting survives instrument preset and PNA application restart.

Parameters

<bool> Single window print state. Chose from:
OFF or (0) - Print up to four windows per page.
ON or (1) - Print only one window per page.

Examples

```
HCOP:ITEM:SWIN 1  
hcopy:item:swindow:state off
```

Query Syntax HCOPY:ITEM:SWINDOW[:STATe]?

Return Type Boolean

Default OFF (0)

HCOPY:ITEM:TIME[:STATe] <bool>

(Read-Write) When ON, prints the PNA computer date and time in the header.

This setting survives instrument preset and PNA application restart.

Parameters

<bool> Time stamp print state. Chose from:
OFF or (0) - Does NOT print time stamp.
ON or (1) - Print time stamp.

Examples

```
HCOP:ITEM:TIME 1  
hcopy:item:time:state off
```

Query Syntax HCOPY:ITEM:TIME:[STATe]?

Return Type Boolean

Default OFF (0)

HCOPY:ITEM:TTABLE[:STATe] <bool>

(Read-Write) When ON, prints the trace attributes table.

This setting survives instrument preset and PNA application restart.

Parameters

<bool> Trace attributes table print state. Chose from:
OFF or (0) - Does NOT print the trace attributes table.
ON or (1) - Print the trace attributes table.

Examples

```
HCOP:ITEM:TTABLE 1  
hcopy:item:ttable:state off
```

Query Syntax HCOPY:ITEM:TTABLE[:STATE]?

Return Type Boolean

Default OFF (0)

HCOPY:ITEM:WFRaction <value>

(Read-Write) Sets the vertical amount of a page that is filled by the measurement windows.

This setting survives instrument preset and PNA application restart.

Parameters

<value> Window size as a fraction of the page. Chose a value from .4 (40%) to 1.0 (100%)

Examples

```
HCOP:ITEM:WFR .8  
hcopy:item:wfraction .5
```

Query Syntax HCOPY:ITEM:WFRaction?

Return Type Numeric

Default .4

HCOPY:ITEM:WINDows[:STATE] <bool>

(Read-Write) When ON, prints measurement windows.

Use [HCOPY:ITEM:AWINDOW](#) to specify all windows or only the active window.

This setting survives instrument preset and PNA application restart.

Parameters

- <bool> Windows print state. Chose from:
- OFF** or (0) - Does not print measurement windows.
 - ON** or (1) - Print measurement windows.

Examples

```
HCOPY:ITEM:WIND 1  
hcopy:item:windows:state off
```

Query Syntax HCOPY:ITEM:WINDows[:STATE]?

Return Type Boolean

Default OFF (0)

HCOPY:PAGE:DIMensions:LLEft <left, lower>

(Read-Write) Sets the left and lower page margins.

This setting survives instrument preset and PNA application restart.

Parameters

- <left> Left page margin as a percentage of entire page width. Value must be between 0 and 1.
- <lower> Lower page margin as a percentage of entire page length. Value must be between 0 and 1.

Examples

```
HCOPY:PAGE:DIM:LLEF .10,.10  
hcopy:page:dimensions:lleft .5,.7
```

Query Syntax HCOPY:PAGE:DIMensions:LLEft?

Return Type Numeric, Numeric

Default Depends on selected page size

HCOPY:PAGE:DIMensions:URIGHt <right, upper>

(Read-Write) Sets the right and upper page margins.

This setting survives instrument preset and PNA application restart.

Parameters

- <right> Right page margin as a percentage of entire page width. Value must be between 0 and 1.
- <upper> Upper page margin as a percentage of entire page length. Value must be between 0 and 1.

Examples

```
HCOPY:PAGE:DIM:URIG .10,.10  
hcopy:page:dimensions:uright .5,.7
```

Query Syntax HCOPY:PAGE:DIMensions:URIGHT?

Return Type Numeric, Numeric

Default Depends on selected page size

HCOPY:PAGE:ORientation <char>

(Read-Write) Sets the page orientation.

This setting survives instrument preset and PNA application restart.

Parameters

<char> Choose from:

- PORTrait
- LANDscape

Examples

```
HCOPY:PAGE:ORI PORT  
hcopy:page:orientation landscape
```

Query Syntax HCOPY:PAGE:ORientation?

Return Type Character

Default PORTrait

HCOPY:PAGE:SIZE <int>

(Read-Write) Sets the paper type, which implies the page size.

This setting survives instrument preset and PNA application restart.

Parameters

<int> Choose from:1

Integer	Description
1	Letter 8 1/2 x 11 in
2	Letter Small 8 1/2 x 11 in
3	Tabloid 11 x 17 in
4	Ledger 17 x 11 in
5	Legal 8 1/2 x 14 in
6	Statement 5 1/2 x 8 1/2 in
7	Executive 7 1/4 x 10 1/2 in
8	A3 297 x 420 mm
9	A4 210 x 297 mm
10	A4 Small 210 x 297 mm
11	A5 148 x 210 mm
12	B4 (JIS) 250 x 354
13	B5 (JIS) 182 x 257 mm

For more paper type choices, see Microsoft's "wingdi.h" file, which can be downloaded as part of the Platform SDK.

Examples

```
HCOPY:PAGE:SIZE 2
```

```
hcopy:page:size 5
```

Query Syntax HCOpy:PAGE:SIZE?

Return Type Integer

Default 1

HCOPY:SDUMp:DATA?

(Read-only) Returns the display image in a definite-length arbitrary binary block. The format of the data is PNG by default. Use [HCOP:SDUMp:DATA:FORMat](#) to change the format.

This command is equivalent to saving an image to the PNA ([HCOPY:FILE](#)) and then using [MMEM:TRAN](#) to transfer the file to the computer.

Examples `HCOP:SDUM?`
`hcopy:sdump?`

Return Type A definite-length arbitrary binary block

Default Not Applicable

HCOPY:SDUMp:DATA:FORMat <char>

(Read-Write) Sets the graphic format for [HCOPY:SDUMp:DATA?](#)

Parameters

<char> Choose from: **JPG** | **BMP** | **PNG**

Examples `HCOP:SDUMp:DATA:FORMat BMP`

Query Syntax `HCOPY:SDUMp:DATA:FORMat?`

Return Type Character

Default PNG

HCOPY:PRINters?

(Read-only) Returns a comma-separated list of printers installed on the PNA. Select a printer using [HCOPY:DPRinter](#).

This setting survives instrument preset and PNA application restart.

Examples `HCOP:PRIN?`
`hcopy:printers?`

Query Syntax `HCOPY:PRINters?`

Return Type String

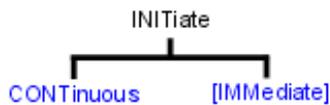
Default Not Applicable

Last modified:

13-Apr- 2011	Obsolete seg data
15-Mar- 2010	Added SDUMP commands (A.09.20)
Nov. 1, 2006	Added new commands

Initiate Commands

Controls triggering signals



Click on a [blue](#) keyword to view the command details.

See Also

- **Example** [Triggering the PNA](#)
 - [Learn about Triggering](#)
 - [Synchronizing the PNA and Controller](#)
 - [SCPI Command Tree](#)
-

INITiate:CONTInuous <boolean>

(Read-Write) Specifies whether the PNA trigger source is set to Internal (continuous) or Manual.

- For SIMPLE, single-triggering of a single channel, use [Sens:Sweep:Mode SINGLE](#) which sets the number of trigger signals each channel will ACCEPT (Continuous, Groups, **Single**, or HOLD - none.)
- This command is a subset of [TRIG:SEQ:SOURce](#), which can also set the trigger source to External.
- See a [map of user interface to SCPI triggering commands](#).
- For more information on triggering, see the [PNA Trigger Model](#).
- **See the Example program:** [Triggering the PNA using SCPI](#).

Parameters

<boolean> **ON** (or 1) - Internal (continuous) trigger.

OFF (or 0) - Manual sweep. Use [INIT:IMMediate](#) to send a trigger signal

Examples

```
INIT:CONT ON
initiate:continuous off
```

Query Syntax INITiate:CONTInuous?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

INITiate<cnum>[:IMMEDIATE]

(Write-only) Stops the current sweeps and immediately sends a trigger. (Same as [Trigger!](#) on the PNA front panel).

- This command requires [Trigger:Source](#) to be set to Manual. This causes ONE trigger signal to be SENT each time INIT:IMM is issued.
- For SIMPLE, single-triggering of a single channel, use [Sens:Sweep:Mode SINGLE](#) which sets the number of trigger signals each channel will ACCEPT (Continuous, Groups, **Single**, or HOLD - none.)

See the Example program: [Triggering the PNA using SCPI](#)

Note: An SMC Fixed Output measurement cannot be triggered using this command. For more information, see the [example program](#).

To trigger ALL channels in turn:

Set ALL channels to Sens<ch>:Sweep:Mode Continuous. The <ch> argument in INIT<ch>:IMM is ignored.

Then...

- TRIG:SCOP ALL triggers ALL channels (in sequence) each time Init:Imm is sent.
- TRIG:SCOP CURRent triggers ONLY the NEXT channel each time Init:Imm is sent.

To trigger ONLY a specified channel:

1. Set ALL channels to Sens<ch>:Sweep:Mode HOLD
2. Send TRIG:SCOP CURRent
3. Send Init<ch>:Imm where <ch> is the channel to be triggered.

Advanced Situations that require some channels to be in CONT and others in HOLD are rare. The following describes the behavior of the Init:Imm command in these situations:

When [Trigger:Scope](#) = Global:

- If the SPECIFIED <cnum> channel is in hold mode, it is put in single trigger (accepts 1 trigger signal) and goes to the end of the queue of channels to be triggered. The other 'non-hold' channels are triggered. The next Init:Imm triggers the specified channel first.

For example: ch1 is in Hold, ch2 and ch3 are in CONT and we send INIT1:IMM

- On the first INIT:IMM, ch2 and ch3 is triggered.
- next INIT:IMM, ch1, ch2, ch3 is triggered.
- next INIT:IMM, ch2 and ch3 is triggered.
- next INIT:IMM, ch1, ch2, ch3 is triggered, and so forth.

When Trigger:Scope = Channel

- Only ONE channel is triggered for each issued INIT<ch>:IMM command.
- If the specified channel is in hold, it is put in single trigger (accepts 1 trigger signal) and goes the end of the queue of channels to be triggered as in the 'Global' example.

This is one of the PNA overlapped commands. [Learn more.](#)

Parameters

<num> Any existing channel number. If unspecified, value is set to 1

Examples

```
INIT
initiate2:immediate
```

Query Syntax Not applicable

Default Not applicable

Last modified:

8-Mar-2013 Updated again with Single note
 April 23, 2007 Updated Init:Imm

Memory Commands

The memory commands control saving and loading instrument states and measurement trace data to the hard drive. To read and write trace data in GPIB format, see [CALC:DATA](#).

MMEMory:

[CATalog?](#)

[CDIRectory](#)

[COPY](#)

[DATE?](#)

[DELete](#)

[LOAD](#)

[MDIRectory](#)

[MOVE](#)

[RDIRectory](#)

[STORE](#)

| [CITI](#)

| [DATA](#)

| [FORMat](#)

| [DATA](#)

| [ENR](#)

| [TRACe](#)

| [CONTents](#)

| [CITIfile](#)

| [FORMat](#)

| [CITIfile](#)

| [SNP](#)

[TIME?](#)

[TRANsfer](#)

Click on a [blue](#) keyword to view the command details.

Red commands are [superseded](#).

See Also

- [Example Programs](#)
- [Learn about Save / Recall and File Types](#)
- [Synchronizing the PNA and Controller](#)

- [SCPI Command Tree](#)

Specifying Path Names

The MMEM commands use the following rules to specify path names:

- The default folder for Windows 7 is "c:\users\public\documents\Network analyzer" [Learn more](#).
 - You can change the active directory using [MMEMory:CDIRectory](#).
 - Specify only the file name if using the active directory.
 - You can also use an absolute path name to specify the folder and file.
-

MMEMory:CATalog[:<char>]? [<folder>]

(Read-only) Returns a comma-separated string of file names that are in the specified folder. If there are no files of the specified type, "NO CATALOG" is returned. [Learn about File Types](#)

Parameters

<char> The type of files to list. Choose from:

- **STAtE** - Instrument states (.sta)
- **CORRection** - Calibration Data (.cal)
- **CSARchive** - Instrument state and calibration data (.csa)
- **CStAtE** - Instrument state and link to Calibration data (.cst)

If unspecified then ALL file types (even unknown types) are listed.

<folder> String - Any existing folder name. See [Specifying Path Names](#)

Examples

```
MMEM:CAT? 'lists all files from the current folder  
mmemory:catalog:correction? c:\users\public\documents\Network  
analyzer\Documents' 'lists .cal files from the specified folder
```

Default Not applicable

MMEMory:CDIRectory <folder>

(Read-Write) Changes the folder name.

Parameters

<folder> Any drive and folder name that already exists.

If the same level as the default path, then no punctuation is required.

```
MMEM:CDIR Service
```

If the new folder is at a different level than the default, use a slash (/) before the folder name and enclose in quotes.

```
mmemory:cdirectory '/automation' 'changes default directory up one level.'
```

You can use an absolute path to specify the new folder.

```
mmemory:cdirectory 'c:/automation/service'
```

Query Syntax MMEMory:CDIRectory? 'Returns the current folder name

Return Type String

Default See [Specifying Path Names](#)

MMEMory:COPY <file1>,<file2>

(Write-only) Copies file1 to file2. Extensions must be specified.

Parameters

<file1> String - Name of the file to be copied. See [Specifying Path Names](#)

<file2> String - Name of the file to be created from file1.

Examples

```
MMEM:COPY 'MyFile.cst','YourFile.cst'
```

Query Syntax Not applicable

Default Not applicable

MMEMory:DATE? <fileName>

(Read-only) Returns the (year, month, day) that the specified file was last saved.

To query the last date and time a cal set was modified, use [CSET DATE?](#) and [CSET:TIME?](#)

See Also

[MMEM:TIME?](#)

Parameters

<fileName> String - File name. See [Specifying Path Names](#)

Example

```
MMEM:DATE? "myFile.txt"

'Returns

+2013,+4,+12

memory:date? "c:\users\public\documents\Network
analyzer\UserCalSets\Calset_18.pcs"

'Returns

+2013,+4,+12
```

Return Type Comma-separated integers

Default Not applicable

MMEMory:DELeTe <file>

(Write-only) Deletes file. Extensions must be specified.

Parameters

<file> String - Name of the file to be deleted. See [Specifying Path Names](#)

Examples

```
MMEM:DEL 'MyFile.cst'
```

Query Syntax Not applicable

Default Not applicable

MMEMory:LOAD[:<char>] <file>

(Write-only) Loads the specified file. [Learn about File Types](#)

Parameters

<char> The type of file to load. Choose from:

- **STATe** - Instrument states (.sta)
- **CORRection** - Calibration Data (.cal)
- **CSARchive** - Instrument state and calibration data (.csa)
- **CState** - Instrument state and link to Calibration data (.cst)
- **ENR** - Excess Noise Source data ([Noise Figure App only](#))
- When <char> is **ENR**, then include **CAL**, - See example below.
- *.sNp files CAN be recalled to the PNA although no <char> is used. See example below.

If <char> is unspecified, the extension must be included in the filename.

If an extension is specified in <file> that does not agree with <char> then no action is taken.

<file> String - Name of the file to be loaded. See [Specifying Path Names](#)

Examples

```
MMEM:LOAD 'MyFile.cst'  
mmemory:load:state 'MyInstState'  
  
MMEM:LOAD:ENR CAL, C:/data/calset/346C_16500.enr"  
  
MMEM:LOAD "MyFile.s2p"
```

Query Syntax Not applicable

Default Not applicable

MMEMory:MDIRectory <folder>

(Write-only) Makes a folder.

Parameters

<folder> String - Name of the folder to make. See [Specifying Path Names](#)

Examples

```
MMEM:MDIR 'MyFolder'  
mmemory:mdirectory 'c:/NewFolder'
```

Query Syntax Not applicable

Default Not applicable

MMEMemory:MOVE <file1>,<file2>

(Write-only) Renames <file1> to <file2>. File extensions must be specified.

Parameters

<file1> String - Name of the file to be renamed. See [Specifying Path Names](#)

<file2> String - Name of the new file.

Examples

```
MMEM:MOVE 'MyFile.cst','YourFile.cst'
```

Query Syntax Not applicable

Default Not applicable

MMEMemory:RDIRECTORY <folder>

(Write-only) Removes the specified folder.

Parameters

<folder> String - Name of the folder to remove. See [Specifying Path Names](#)

Examples

```
MMEM:RDIR 'MyFolder'
```

Query Syntax Not applicable

Default Not applicable

MMEMemory:STORE[:<char>] <file>

(Write-only) Stores the specified file (.sta, .cal, .cst, .csa, .snp, s2px).

Learn about [saving SNP files on the PNA](#).

Learn about [saving S2Px files on the PNA](#).

To save other data files, use [MMEM:STOR:DATA](#).

To save ENR files, use [MMEMory:STORe:ENR](#)

Parameters

<char> Optional argument. The type of file to store. Choose from:

- **STATe** - Instrument states (.sta)
- **CORRection** - Calibration Data (.cal)
- **CSARchive** - Instrument state and calibration data (.csa)
- **CSTate** - Instrument state and link to Calibration data (.cst)

No <char> is specified for s1p, s2p, s2px and so forth.

Include either <char> or the file extension. If both <char> and the extension are specified, they must agree or an error is returned and no action is taken. See examples below.

[Learn about File Types](#)

<file> String - Name of any valid file that does not already exist. See [Specifying Path Names](#)

Examples

```
MMEM:STOR:STAT 'myState'  
  
mmemory:store 'c:/bin/myState.sta'  
  
MMEM:STOR 'MyData.S2P'
```

Query Syntax Not applicable

Default Not applicable

MMEMory:STORe:CITifile:DATA <filename> - Superseded

This command is replaced with [MMEMory:STORe:DATA](#).

(Write only) Saves UNFORMATTED trace data to .cti file. [Learn more.](#)

Parameters

<filename> Any path that already exists with filename.

If the same level as the default, then no path is required.

```
MMEM:STOR:CIT:DATA 'MYFile.cti'
```

Of you can specify an absolute path and filename:

```
mmemory:store:citifile:data "c:\users\public\documents\Network analyzer\myFile.cti'
```

Query Syntax Not Applicable

Default See [Specifying Path Names](#)

MMEMory:STORe:CITifile:FORMat <filename> - Superseded

This command is replaced with [MMEMory:STORe:DATA](#).

(Write only) Saves FORMATTED trace data to .cti file. [Learn more.](#)

Parameters

<filename> Any path that already exists with filename.

If the same level as "c:\users\public\network analyzer\documents", then no path is required

```
MMEM:STOR:CIT:FORM 'MYFile.cti'
```

Of you can specify an absolute path and filename:

```
mmemory:store:citifile:format "c:\users\public\documents\Network analyzer\myFile.cti'
```

Query Syntax Not Applicable

Default See [Specifying Path Names](#)

MMEMory:STORe:DATA <filename>,<type>,<scope>,<format>,<selector>

(Write-only) Stores trace data to the following file types: *.prn, *.cti, *.csv, *.mdf

To save snp files, use [Calc:Data:SNP:PORTs:SAVE](#)

To save state and calibration files, use [MMEM:STORE](#)

This command replaces the following:

- [MMEMory:STORE:CITifile:DATA](#)
- [MMEMory:STORE:CITifile:FORMat](#)
- [MMEMory:STORE:TRACe:FORMat:CITifile](#)
- [MMEMory:STORE:TRACe:CONTent:CITifile](#)

NOTE: Not all choices are valid with other arguments. See [Valid parameter combinations](#) below.

Parameters

<filename> (String) Name and extension of the file to which data will be saved. If the extension does not agree with the file type, an error is NOT returned but the data may NOT be what you expect.

[See rules for specifying a filename.](#)

<type> (String) File type to save. Choose from:

"PRN Trace Data" - *.prn data. [Learn more.](#)

"Citifile Data Data" - unformatted *.cti data. [Learn more.](#)

"Citifile Formatted Data" - formatted *.cti data.

"CSV Formatted Data" - formatted *.csv data. [Learn more.](#)

"MDIF Data" - *.mdf data. [Learn more.](#)

"GCA Sweep Data" - Gain compression data. [Learn more.](#)

"IMD Sweep Data" - Swept IMD data. [Learn more.](#)

<scope> (String) How much data to save. Choose from:

"Trace" - only the specified measurement number is saved.

"Displayed" - all displayed measurements are saved.

"Channel" - all measurements that are in the channel in which the selected measurement reside are saved.

"Auto"

For all Standard Meas Class (S-parameter) channels:

- When correction is OFF, the specified trace is saved.
- When correction is ON, all corrected parameters associated with the calibrated ports in the Cal Set are saved.

For all other channels:

- When correction is OFF or ON, the specified trace is saved.

<format> The format in which data is saved. Choose from:

"Displayed" - the format is the same as that in which it is displayed on the PNA screen.

"RI" - Real / Imaginary

"MA" - Magnitude / Angle

"DB" - LogMag / Degrees

<selector> (Integer) Choose from:

-1 Use when <scope> = "Displayed" (does NOT require a selected trace).

<measurement number> Use for all other <scope> selections. Use [Calc:Par:MNUM?](#) to read the measurement number of the selected trace.

The following are **valid parameter combinations** for ALL measurement classes:

Parameters			
<type> (String)	<scope> (String)	<format> (String)	<selector> (Numeric)
"PRN Trace Data"	"Trace"	"Displayed"	Measurement number
Example: MMEMemory:STORe:DATA "myData.prn","PRN Trace Data","Trace","Displayed",2			
"Citifile Data Data"	"Trace" or "Auto" or "Channel"	"RI"	Measurement number
	"Displayed"	"RI"	-1
Example: MMEMemory:STORe:DATA "myData.cti","Citifile Data Data","AUTO","RI",3			
	"Trace" or "Auto"	"RI" or "MA" or "DB"	Measurement number

"Citifile Formatted Data"	"Channel"	"RI" or "MA" or "DB" or "Displayed"	Measurement number
	"Displayed"	"RI" or "MA" or "DB" or "Displayed"	-1

Example: `MMEMemory:STORe:DATA "myData.cti","Citifile Formatted Data","AUTO","MA",3`

"CSV Formatted Data"	"Trace" or "Auto" or "Channel"	"RI" or "MA" or "DB" or "Displayed"	Measurement number
	"Displayed"	"RI" or "MA" or "DB"	-1

Example: `MMEMemory:STORe:DATA "myData.csv","CSV Formatted Data","displayed","RI",-1`

"MDIF Data"	"Trace" or "Auto"	"RI" or "Displayed" or "Channel"	Measurement number
	"Displayed"	"RI" or "Displayed"	-1

Example: `MMEMemory:STORe:DATA "myData.mdf","MDIF Data","displayed","displayed",-1`

The following parameter combinations save *.csv files in specific formats for GCA and Swept IMD classes:

Parameters			
<type> (String)	<scope> (String)	<format> (String)	<selector> (Numeric)

"GCA Sweep Data"	"Auto"	"DB"	GCA channel number
------------------	--------	------	--------------------

Example: `MMEMemory:STORe:DATA "myData","gca sweep data","displayed","displayed",-1`

"IMD Sweep Data"	"Auto"	"DB"	Swept IMD channel number
------------------	--------	------	--------------------------

Example: `MMEMemory:STORe:DATA "myData.mdf","MDIF Data","displayed","displayed",-1`

Query Syntax Not applicable

Default Not applicable

MMEMemory:STORe:ENR CAL, <file>

(Write-only) Stores an **ENR** (Excess Noise Source) data. ([Noise Figure App only](#))

To set and read ENR data, use [SENS:CORR:ENR:CAL:TABLE:DATA](#).

Parameters

<file> String - Name of any valid file that is not already in existence. See [Specifying Path Names](#)

Examples `MMEM:STOR:ENR CAL, "C:/data/calset/346C_16500.enr"`

Query Syntax Not applicable

Default Not applicable

MMEMory:STORe:TRACe:FORMat:CITifile <char> - Superseded

This command is replaced with [MMEMory:STORe:DATA](#).

(Read-Write) Specifies the format of subsequent citifile save statements.

Parameters

<char> Format in which the citifile will be saved with subsequent [MMEMory:STORe:CIT:FORMat](#) statements. Choose from:

MA - Linear Magnitude / degrees

DB - Log Magnitude / degrees

RI - Real / Imaginary

AUTO - Format in which the trace is already displayed. If other than Log Mag, Linear Magnitude, or Real/Imag, then the format will be in Real/Imag.

DISP - Displayed format.

Examples `MMEM:STOR:TRAC:FORM:CIT MA`

Query Syntax `MMEMory:STORe:TRACe:FORMat:CITifile?`

Return Type Character

Default Auto

MMEMory:STORe:TRACe:CONTents:CITifile <char> - Superseded

This command is replaced with [MMEMory:STORe:DATA](#).

(Read-Write) Specifies the contents of subsequent citifile save statements. (See [Data Define Saves](#))

Parameters

<char> Choose from:

SING - Single trace

DISP - All displayed traces

AUTO - All displayed traces

Examples

```
MMEM:STOR:TRAC:CONT:CIT SING
```

Query Syntax MMEMory:STORe:TRACe:CONTents?

Return Type Character

Default Auto

MMEMory:STORe:TRACe:FORMat:SNP <char>

(Read-Write) Specifies the format of subsequent .s1p, .s2p, .s3p; s4p save statements. [Learn more](#).

To save SNP data, use [CALC:DATA:SNP:PORTs:SAVE](#)

Parameters

<char> Choose from:

MA - Linear Magnitude / degrees

DB - Log Magnitude / degrees

RI - Real / Imaginary

AUTO - data is output in currently selected trace format. If other than LogMag, LinMag, or Real/Imag, then output is in Real/Imag.

Examples

```
MMEM:STOR:TRAC:FORM:SNP MA
```

Query Syntax MMEMory:STORe:TRACe:FORMat:SNP?

Return Type Character

Default Auto'

MMEMory:TIME? <fileName>

(Read-only) Returns the (hour, minute, second) that the specified file was last saved. The time is returned in local time as setup in the PNA operating system.

To query the last date and time a cal set was modified, use [CSET DATE?](#) and [CSET:TIME?](#)

See Also

[MMEM:DATE?](#)

Parameters

<fileName> String - File name. See [Specifying Path Names](#)

Example

```
MMEM:TIME? "myFile.txt"

'returns

+12,+34,+12

mmemory:time? "c:\users\public\documents\Network
analyzer\UserCalSets\Calset_18.pcs"

'returns

+12,+34,+12
```

Return Type Comma-separated integers

Default Not applicable

MMEMory:TRANSfer <fileName>,<dataBlock>

(Read-Write) Transfers data between the PNA and an external controller. Other MMEM commands transfer data between the PNA application and the PNA hard drive. If <fileName> already exists, it will be overwritten. The file must be no larger than 20MB.

To read **trace data** from the PNA in block format, use [CALC:DATA](#).

Parameters

<fileName> String - File name. See [Specifying Path Names](#)

<dataBlock> [Block Data](#) - The contents of the file.

The data block is a block of binary data. Use the following syntax:

```
#<num digits><byte count><data bytes><NL><END>
```

where:

<**num_digits**> specifies how many digits are contained in <byte_count>

<**byte_count**> specifies how many data bytes will follow in <data bytes>

Example [See example program](#)

Query Syntax MMEMory:TRANsfer? <fileName>

Reads block data from the specified file location.

Default Not applicable

Last modified:

- 14-Jan-2014 Added link to mmem:store:enr
Added argument to Store command (SW)
Modified default path (Win7)
Added date, time commands
Fixed examples in superseded commands
Added back snp to MMEM:STOR
Updated for csv and mdif formats (9.2)
Fixed noise keywords for load and store.
Added noise keywords to load and store.
- 9/12/06 MQ Store command has reference to PORTS Snp.

Output Commands

Controls two output functions: RF power and Noise Source.

OUTPut:

| [MANual:NOISe\[:STATe\]](#)

| [\[:STATe\]](#)

Click on a [blue](#) keyword to view the command details.

See Also

- [Example Programs](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

OUTPut:MANual:NOISe[:STATe] <bool>

(Read-Write) Sets and reads the noise source ([28V](#)) ON or OFF.

Parameters

<bool> **ON (1)** - Noise source ON

OFF (0) - Noise source OFF

Examples

```
OUTP:MAN:NOIS 0  
output:manual:noise:state 1
```

Query Syntax OUTPut:MANual:NOISe[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default For PNA models with a [Noise Figure option](#) (028/029/H29), the 28V line is always ON. The ON/OFF state is also available from a PNA softkey menu.

For PNA models WITHOUT a Noise Figure option (028/029/H29), the 28V line is OFF by default and survives a preset. The ON/OFF state is NOT available from a PNA softkey menu.

OUTPut[:STATe] <ON | OFF>

(Read-Write) Turns RF power from the source ON or OFF.

[See note about source power state with instrument state save and recall.](#)

Parameters

<ON | OFF> **ON** (or 1) - turns RF power ON

OFF (or 0) - turns RF power OFF

Examples

```
OUTP ON  
output:state off
```

Query Syntax OUTPut[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

Last Modified:

24-Feb-2009 Replaced True/False

22-Aug-2007 Added noise command

Route Command

[Learn about Frequency Offset](#)

[SCPI Command Tree](#)

ROUTE<cnum>:PATH:LOOP[:R1] <char>

(Read-Write) Throws internal switch to reference receiver when the specified channel is measured.

N523xA models do NOT have the R1 reference receiver switch.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<char> Position of the switch. Choose from:

INTernal - bypass R1 Loop. Connects the port 1 source directly to the R1 receiver.

EXTernal - flow through R1 Loop. Allows direct access to the R1 receiver through the Reference 1 front-panel connectors.

Examples

```
ROUT:PATH:LOOP INT
route2:path:loop:r1 external
```

Query Syntax ROUTE<cnum>:PATH:LOOP:R1?

Return Type Character

Default INTernal

Last Modified:

30-Apr-2009 Added PNA-X

Sense:Average Commands

Sets sweep-to-sweep averaging parameters. Averaging is a noise reduction technique that averages each data point over a user-specified number of sweeps. Averaging affects all of the measurements in the channel.

SENSe:AVERage
CLEar
COUNT
MODE
[STATe]

Click on a [blue](#) keyword to view the command details.

See Also

- [Example](#) using some of these commands.
- [Learn about Averaging](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

SENSe<cnum>:AVERage:CLEar

(Write-only) Clears and restarts averaging of the measurement data. Does NOT apply to point averaging.

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1.

Examples

```
SENS:AVER:CLE  
sense2:average:clear
```

Query Syntax Not applicable

Default Not applicable

SENSe<cnum>:AVERage:COUNT <num>

(Read-Write) Sets the number of measurements to combine for an average. Must also set [SENS:AVER\[:STATe\] ON](#)

Parameters

- <num> Any existing channel number; if unspecified, value is set to 1.
- <num> Number of measurements to average. Choose any number between **1** and **65536** (2^{16}).

Examples

```
SENS:AVER:COUN 999  
sense2:average:count 73
```

Query Syntax SENSE<num>:AVERage:COUNT?

Return Type Numeric

Default 1

SENSe<num>:AVERage:MODE <char>

(Read-Write) Sets the type of averaging to perform: Point or Sweep.

Parameters

- <num> Any existing channel number; if unspecified, value is set to 1.
- <num> Averaging Type. Choose from:
 - POINT** - Averaging measurements are made on each data point before stepping to the next data point. (Not available on 'C' models).
 - SWEEP** - Averaging measurements are made on subsequent sweeps until the required number of averaging sweeps are performed.

Examples

```
SENS:AVER:MODE POIN  
sense2:average:mode sweep
```

Query Syntax SENSe<num>:AVERage:MODE?

Return Type Character

Default Sweep

SENSe<num>:AVERage[:STATe] <ON | OFF>

(Read-Write) Turns trace averaging ON or OFF.

Parameters

- <cnum> Any existing channel number; if unspecified, value is set to 1.
- <ON | OFF> **ON** (or 1) - turns averaging ON.
OFF (or 0) - turns averaging OFF.

Examples

```
SENS:AVER ON  
sense2:average:state off
```

Query Syntax SENSE<cnum>:AVERage[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default Off

Last Modified:

- June 8, 2009 Not available on 'C' models
- 13-Oct-2008 Added Point averaging

Sense:Bandwidth Commands

SENSe:BANDwidth:

RESolution <num>

TRACk <bool>

See Also

- [Example Programs](#)
- [Learn about IF Bandwidth](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

SENSe<cnum>:BANDwidth | BWIDth[:RESolution] <num>

(Read-Write) Sets the bandwidth of the digital IF filter to be used in the measurement. (Use either **Sense:Bandwidth** or **Sense:Bwidth**)

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<num> IF Bandwidth in Hz. The list of valid IF Bandwidths is different depending on the PNA model. [\(Click to see the lists.\)](#) If an invalid number is specified, the analyzer will round up to the closest valid number.

This parameter supports MIN and MAX as arguments. [Learn more.](#)

Examples

```
SENS:BWID 1KHZ
sense2:bandwidth:resolution 1000
```

Query Syntax SENSe<cnum>:BANDwidth | BWIDth[:RESolution]?

Return Type Numeric

Default Varies with PNA model.

SENSe<cnum>:BANDwidth | BWIDth:TRACk <bool>

(Read-Write) Sets and returns the state of the [Reduce IF BW at Low Frequencies](#) feature.

(Use either **Sense:Bandwidth:Track** or **Sense:Bwidth:Track**).

Parameters

<cnun> Any existing channel number. If unspecified, value is set to 1

<bool> Choose from:

ON or **1** - Reduce IF BW at Low Frequencies is set ON

OFF or **0** - Reduce IF BW at Low Frequencies is set OFF

Examples

```
SENS:BWID:TRAC OFF
sense2:bandwidth:track 1
```

Query Syntax SENSE<cnun>:BANDwidth | BWIDth:TRACk?

Return Type Boolean

Default ON

Last Modified:

7-May-2012 Removed Preset

15-Jan-2008 MIN and MAX added

Sense:Class Command

SENSe:CLASs:

NAME?

Click on a [blue](#) keyword to view the command details.

See Also

- [Learn about Measurement Class](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

SENSe<num>:CLASs:NAME?

(Read-only) Returns the measurement class name of the specified channel. Use [Calc:Custom](#) commands to create measurements from classes other than standard S-Parameters. Use [Cal:Par:Define:Ext](#) to create standard measurements.

Parameters

<num> Any existing channel number; if unspecified, value is set to 1.

Examples

```
SENS:CLAS:NAME?  
sense2:class:name?
```

```
For a standard S-Parameter channel, returns...  
"Standard"
```

Default Not applicable

Last Modified:

12-May-2008 MX New topic

Sense:Correction Commands

Performs and applies calibration and other error correction features.

- To perform a Guided Calibration, use ONLY the [Sens:Corr Coll:GUIDed](#) commands.
- To perform an Unguided Calibration, do NOT use the Sens:Corr:Coll:Guided commands.
- See the ["Unguided" example programs](#) for clarification.

SENSe:CORRection

CCHeck

| [\[ACQuire\]](#)

| [DONE](#)

| [PARAmeter](#)

CKIT - More Commands

COLLect

| [\[ACQuire\]](#)

| [APPLy](#)

| [CKIT - More Commands](#)

| [DISPlay:WINDow](#)

| [AOFF](#)

| [\[STATe\]](#)

| [GUIDed - More Commands](#)

| [ISOLation:](#)

| [AVER:INCRement](#)

| [ECAL\[::STATe\]](#)

| [METHod](#)

| [NOISe](#)

| [ENR:ADAP:DEEMbed\[::STATe\]](#)

| [LO:PCAL\[::STATe\]](#)

| [PSEN:ADAP:DEEMbed\[::STATe\]](#)

| [SAVE](#)

| [SESSion - More Commands](#)

| [SWEep:CHANnel](#)

| [AOFF](#)

| [\[STATe\]](#)

CSET - More Commands

ENR:CALibration:TABLE

| [DATA](#)

| [ID:DATA](#)

| [SERial:DATA](#)

EXTension - More Commands

GCSetup

| [POWER](#)

| **SENSor:**

| [CKIT](#)

| [CONNector](#)

IMD - More Commands

IMPedance:INPut

| [MAGNitude](#)

INTerpolate[:STATe]

ISOLation[:STATe]

METHods:MATCh

PREFerence

| **CALibration**

| [\[FOM:\]RANGe](#)

| **CSET**

| [SAVE](#)

| **SAVUser**

| **ECAL**

| [ORientation](#)

| [PMAP](#)

| [SIMCal](#)

| [TRIG:FREE](#)

RPOWER:OFFSet

| [\[AMPLitude\]](#)

RVELocity

| [COAX](#)

SFOward

| [\[STATe\]](#)

[\[STATe\]](#)

TCOLd:USER:VALue

TSTandards

| [\[STATe\]](#)

TYPE

| [CATalog?](#)

Click on a [blue](#) keyword to view the command details.

Red commands are superseded.

See Also

- [Example Programs](#)

- New [See Calibrating the PNA Using SCPI](#)
- [Learn about Measurement Calibration](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

SENSe<cnum>:CORRection:CCHeck[:ACQuire] <mod>[,char]

(Write-only) Reads the 'confidence data' associated with the specified ECal module and puts it into memory. The measurement is selected using [SENS:CORR:CCH:PAR](#). This command is compatible with *OPC.

Note: A confidence check can NOT be performed remotely from User Characterizations that are stored on the PNA disk.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1.

<mod> ECAL Module that contains the confidence data. Choose from:

ECAL1

..through..

ECAL50

[char] Optional argument. Specifies which characterization within the ECal module that the confidence data will be read from.

CHAR0 Factory characterization (data that was stored in the ECal module by Agilent). Default if not specified.

CHAR1 User characterization #1

CHAR2 User characterization #2

...and so forth up to:

CHAR12 User characterization #12

Examples

```
SENS:CORR:CCHeck ECAL2
```

```
sense2:correction:ccheck:acquire ecall,char1
```

Query Syntax Not applicable

Default Not applicable

SENSe<num>:CORRection:CCHeck:DONE

(Write-only) Concludes the Confidence Check and sets the ECal module back into the idle state.

Parameters

<num> Any existing channel number. If unspecified, value is set to 1

Examples

```
SENS:CORR:CCH:DONE  
sense2:correction:ccheck:done
```

Query Syntax Not applicable

Default Not applicable

SENSe<num>:CORRection:CCHeck:PARAmeter <Mname>

(Read-Write) Specifies an existing measurement to be used for the Confidence Check.

Note: A confidence check can NOT be performed remotely from User Characterizations that are stored on the PNA disk.

Parameters

<num> Any existing channel number. If unspecified, value is set to 1

<Mname> Name of the measurement you are selecting for the confidence check. The measurement must already exist.

Examples

```
SENS:CORR:CCH:PAR 'TEST'  
'selects the measurement "test" on channel 1 for the confidence  
check  
  
sense2:correction:ccheck:parameter 'test'  
'selects the measurement "test" on channel 2 for the confidence  
check
```

Query Syntax SENSe<num>:CORRection:CCHeck:PARAmeter?

Returns the name of the selected measurement on channel <num>.

Return Type String

Default Not applicable

SENSe<num>:CORRection:COLLect[:ACQuire] <class>[,subclass][,sync]

(Write-only) For UNGUIDED calibration, measures the specified standards from the selected calibration kit. The calibration kit is selected using the [Sense:Correction:Collect:CKIT](#) command.

For using two sets of standards, see [SENS:CORR:TST](#).

Note: Before using this command you must select two items:

1. Select a calibration method using [SENS:CORR:COLL:METH](#)
2. Select a measurement using [CALC:PAR:SEL](#). You can select one measurement for each channel.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<class> **Measures the standards associated with these class labels.** Choose from:

Label	Forward	Reverse
STAN1	S11A	S22A
STAN2	S11B	S22B
STAN3	S11C	S22C
STAN4	S21T	S12T
STAN5	Generic Isolation; not associated with calibration kit definition.	
ECAL1		

through ECAL modules

ECAL50

RESPonse Same as [Normalize](#) selection in Unguided Cal. (subclass is ignored)

POWER Take a receiver power cal sweep and turn correction ON

SLSET Sets 'sliding load type', and increments the "number of slides" count. The total number of slides is critical to the correct calculation of the sliding load algorithm. See a [sliding load cal example](#).

SLDONE Computes the sliding load using a circle fit algorithm.

[subclass] Optional argument. For mechanical calibration kits, choose from the following to specify the standard to be acquired from the [SENS:CORR:COLL:CKIT:ORDER](#) list. If not specified, subclass is set to **SST1**.

- SST1** First standard in the order list
- SST2** Second standard in the order list
- SST3** Third standard in the order list
- SST4** Fourth standard in the order list
- SST5** Fifth standard in the order list
- SST6** Sixth standard in the order list
- SST7** Seventh standard in the order list

If an ECAL module (1 through 8) is specified for <class>, choose one of the following for specifying which characterization within the ECAL module will be used for the acquire. If not specified, the default is **CHAR0**.

- CHAR0** Factory characterization (data that was stored in the ECAL module by Agilent)
- CHAR1** User characterization #1
- CHAR2** User characterization #2

...and so forth up to:

- CHAR12** User characterization #12

[sync] Optional argument. Choose from:

SYNchronous - blocks SCPI commands during standard measurement (default behavior)

ASYNchronous - does NOT block SCPI commands during standard measurement.

[Learn more about this argument](#)

Examples

```
SENS:CORR:COLL STAN1

'If SENS:CORR:COLL:CKIT:ORDER2 5,3,7
was specified, the following command measures standard 3 (the
second in the order list)
sense1:correction:collect:acquire stan3,sst2

SENS:CORR:COLL ECAL4,ASYN; *OPC?

sense2:correction:collect:acquire ecal2,char1
```

Query Syntax Not applicable

Default Not applicable

SENSe<cnum>:CORRection:COLLect:APPLY

(Write-only) Applies error terms to the measurement that is selected using [Calc:Par:Select](#).

Note: Before using this command you must select a measurement using [CALC:PAR:SEL](#). You can select one measurement for each channel.

Note: This command is only necessary if you need to modify error terms. If you do not need to modify error terms, [SENSe<cnum>:CORRection:COLLect:SAVE](#) calculates and then automatically applies error terms after you use [SENS:CORR:COLL:ACQuire](#) to measure cal standards.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

Example

```
1. CALCulate2:PARAmeter:SElect S21_2 'select the measurement to
   apply terms to
2. SENSe2:CORRection:COLLect:MEthod SPARSOLT 'set type of cal
   method.
3. CALCulate2:DATA? SCORR1 'download the error term of interest
4. 'Modify the error term here
5. CALCulate2:DATA SCORR1 'upload the error term of interest
6. SENSe2:CORRection:COLLect:APPLY 'applies the error terms to
   the measurement
```

Query Syntax Not applicable

Default Not applicable

SENSe:CORRection:COLLect:DISPlay:WINDow:AOff

(Write-only) Clears the flags for windows to be shown during calibrations. To flag a window to be shown see [SENS:CORR:COLL:DISP:WIND](#).

Examples

```
SENS:CORR:COLL:DISP:WIND:AOff
sense:correction:collect:display>window:aoff
```

[See an example using this command.](#)

Query Syntax Not Applicable

Default Not Applicable

SENSe:CORRection:COLLect:DISPlay:WINDow<wNum>[:STATe] <bool>

(Write-only) Set the 'show' state of the window to be displayed during a calibration. [Learn more](#).

When this command is sent, the specified window is 'flagged' to be shown during calibration. The flag is cleared when the window is closed. A Preset or Instrument State Recall also closes the window. If the same window number is reopened, this command must be sent again to show the window during a calibration. The flag is NOT saved with an instrument state.

Send this command for each additional window to show during a calibration.

Parameters

<wNum> Window number to show during a calibration. The calibration window will also be shown with this window.

The window must already be created.

Use [DISPlay:CATalog?](#) to read all existing window numbers.

<bool> Window state. Choose from:

ON (or 1) - Show the specified window during calibration.

OFF (or 0) - Do NOT show the specified window during calibration.

Examples

```
SENS:CORR:COLL:DISP:WIND1 1
```

```
sense:correction:collect:display>window2:state off
```

[See an example using this command.](#)

Query Syntax Not Applicable

Default OFF

SENSe:CORRection:COLLEct:ISOLation:AVERage:INCRement <num>

(Read-Write) Specifies amount to increment (increase) the channel averaging factor during isolation measurement of the ECal module during an unguided ECal calibration.

Note: if the channel currently has averaging turned OFF and <num> is greater than 1, averaging will be turned ON only during the isolation measurements and with the averaging factor equal to <num>.

Parameters

<num> Incremental Averaging factor. The maximum averaging factor is 65536 (2^{16}).

Examples

```
SENS:CORR:COLL:ISOL:AVER:INCR 16
```

```
sense:correction:collect:isolation:average:increment 0
```

Query Syntax SENSE:CORRection:COLLect:ISOLation:AVERage:INCRement?

Return Type Numeric

Default 8 - If this command is NOT sent, but [ECal isolation is measured](#), then averaging will be turned ON with factor set to 8 during the isolation measurement.

SENSe<cnum>:CORRection:COLLect:ISOLation:ECAL[:STATe] <bool>

(Read-Write) Specifies whether or not the isolation state of the ECal module will be measured as part of an unguided ECal calibration.

An unguided calibration is performed using the SENS:CORR:COLL:METH and SENS:CORR:COLL:ACQ commands.

Note: The inherent isolation of the PNA is better than that attained with this command. ONLY use this command when using an external test set, and ONLY using a 8509x ECal module.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<bool> **ON** (or 1) - isolation is measured during the unguided ECal calibration.

OFF (or 0) isolation is NOT measured during the unguided ECal calibration.

Examples

```
SENS1:CORR:COLL:ISOL:ECAL ON
```

```
sense2:correction:collect:isolation:ecal:state 0
```

Query Syntax SENSE:CORRection:COLLect:ISOLation:ECAL:STATe?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

SENSe<num>:CORRection:COLLect:METhod <char>

(Read-Write) For UNGUIDED calibration, sets the calibration method (also known as 'Calibration Type' on calibration dialog box.) To select a Cal Type from a Cal Set, use [CALC:CORR:TYPE](#).

Note: Before using this command you must select a measurement using [CALC:PAR:SEL](#). You can select one measurement for each channel.

Parameters

<num> Any existing channel number. If unspecified, value is set to 1

<char> Choose from:

Method	Description
NONE	No Cal method
REFL1OPEN	Response Open
REFL1SHORT or REFL1	Response Short
REFL3	Full 1 port
RESPonse	Same as Normalize selection in Unguided Cal.
RPOWER	Receiver Power Cal - Used only with receiver measurements.
TRAN1	Response Thru - Requires a Thru standard.
TRAN2	Response Thru and Isolation - Requires a Thru standard.
SPARSOLT	Full SOLT 2 port
SPARSOLT3	Full SOLT 3 port
SPARTRL	TRL Cal (Delta Match Cal may be required)

Examples

```
SENS:CORR:COLL:METH REFL1  
sense2:correction:collect:method sparsolt
```

Query Syntax SENSe<num>:CORRection:COLLect:METhod?

Return Type Character

Default Not Applicable

SENSe<ch>:CORRection:COLLect:NOISe:ENR:ADAPter:DEEMbed:[STATe] <bool>

(Read-Write) Set and read the state of ENR Adapter de-embedding. [Learn more.](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <bool> ENR Adapter de-embed state. Choose from:
 - OFF** or **0** - Do not force de-embedding.
 - ON** or **1** - Force de-embedding.

Examples

```
SENS:CORR:COLL:NOIS:ENR:ADAP:DEEM 0  
sense2:correction:collect:noise:enr:adapter:deembed:state ON
```

Query Syntax SENSE:CORRection:COLLect:NOISe:ENR:ADAPter:DEEMbed:[STATe]?

Return Type Boolean

Default 0 - OFF

SENSE<ch>:CORRection:COLLect:NOISe:LO<n>:PCAL[:STATe] <bool>

(Read-Write) Enables and disables LO power calibration for NFX.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> LO Stage (number). Choose 1 for NFX.
- <bool> LO Power Cal state. Choose from:
 - OFF** or **0** - Disable LO Power Cal
 - ON** or **1** - Enable LO Power Cal

Examples

```
SENS:CORR:COLL:NOIS:LO1:PCAL 0  
sense2:correction:collect:noise:lo1:pcal:state ON
```

Query Syntax SENSE:CORRection:COLLect:NOISe:LO<n>:PCAL:STATe?

Return Type Boolean

Default 0 - OFF

SENSE<ch>:CORRection:COLLect:NOISe:PSENSor:ADAPter:DEEMbed:[STATe] <bool>

(Read-Write) Set and read the state of power sensor adapter de-embedding. [Learn more.](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <bool> Power sensor adapter de-embed state. Choose from:
- OFF** or **0** - Do not force de-embedding.
 - ON** or **1** - Force de-embedding.

Examples

```
SENS:CORR:COLL:NOIS:PSEN:ADAP:DEEM 0
```

```
sense2:correction:collect:noise:psensor:adapter:deembed:state ON
```

Query Syntax SENSE:CORRection:COLLect:NOISe:PSENsor:ADAPter:DEEMbed:[STATe]?

Return Type

Return Boolean

Type

Default 0 - OFF

SENSe<cnum>:CORRection:COLLect:SAVE

(Write-only) For UNGUIDED calibrations ONLY. This command does the following:

- calculates the error terms using the selected :METHod
- applies the error terms to the selected measurement (turns error correction ON.)
- saves the calibration error-terms to the channels Cal Register or a User Cal Set.

The Cal Register or User Cal Set is determined by the setting of the [SENSe:CORR:PREFeRence:CSET:SAVE](#) command.

Do NOT use this command during an ECAL. When performing an ECAL calibration using [SENSe:CORR:COLL:ACQuire](#), this SAVE operation is performed automatically before the completion of a successful ACQuire.

Before using this command you must select a measurement using [CALC:PAR:SEL](#). You can select one measurement for each channel.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1

Examples

```
SENS:CORR:COLL:SAVE  
sense2:correction:collect:save
```

Query Syntax Not applicable**Default** Not applicable**SENSe:CORRection:COLLect:SWEep:CHANnel:AOff**

(Write-only) Clears ALL flags for channels to sweep during calibration. To flag a channel, see [SENS:CORR:COLL:SWE:CHAN](#).

Examples

```
SENS:CORR:COLL:SWE:CHAN:AOff  
sense:correction:collect:sweep:channel:aoff
```

[See an example using this command.](#)

Default Not applicable**SENSe<num>:CORRection:COLLect:SWEep:CHANnel<num2>[:STATe] <bool>**

(Write-only) Specifies the channel to sweep during a Calibration.

When this command is sent, the <num2> channel is 'flagged' to be swept during calibration.

The flag is cleared when the channel is deleted, if the Measurement Class is changed, or if all measurements are deleted from the channel.

If the same channel number is recreated, this command must be sent again to sweep the channel during a calibration. The flag is NOT saved with an instrument state.

A Preset or Instrument State Recall deletes the channel.

Parameters

<num> The channel to be calibrated. If unspecified, value is set to 1.

<num2> The channel to sweep when waiting to measure a standard.

This channel must already exist with at least one measurement in the channel. If this channel is in continuous sweep mode, it must have the same attenuator settings and path configuration (PNA-X only).

<bool> Channel sweep state. Choose from:

ON (or 1) - Sweep the channel during calibration.

OFF (or 0) - Do NOT sweep the channel during calibration.

Examples

```
SENS:CORR:COLL:SWE:CHAN2 1
```

```
sense2:correction:collect:sweep:channel3:state off
```

[See an example using this command.](#)

Query Syntax Not Applicable

Default OFF

SENSe:CORRection:ENR:CALibration:TABLE:DATA <freq, value, freq, value...>

(Read-Write) Set and read the ENR calibration data. All of the frequency and ENR data must be sent at the same time. Use [MMEM:LOAD](#) to load, and [MMEM:STORE:ENR CAL](#) to save ENR table data from disk. [Learn more about Noise Source ENR files.](#)

Parameters

<freq, value> (Numeric) ENR data. Frequency value in Hz followed by a ENR noise value in dB. Enter as many pairs as necessary.

Examples

```
SENS:CORR:ENR:CAL:TABL:DATA 1.0E9,14.37,2.5E9,15.28
```

```
sense:correction:enr:calibration:table:data
```

```
1.0E9,14.37,2.5E9,15.28
```

Query Syntax SENSe:CORRection:ENR:CALibration:TABLE:DATA?

Return Type Comma separated numeric values

Default Not Applicable

SENSe:CORRection:ENR:CALibration:TABLE:ID:DATA <id>

(Read-Write) Sets and returns ID of ENR table. While this is for informational purposes only, it can be used to record the model of the noise source. [Learn more about ENR files.](#)

Parameters

<id> (String) Identifier for the ENR table.

Examples

```
SENS:CORR:ENR:CAL:TABL:ID:DATA "346C"  
sense:correction:enr:calibration:table:id:data "ENR Table"
```

Query Syntax SENSE:CORRection:ENR:CALibration:TABLE:ID:DATA?

Return Type String

Default Not Applicable

SENSe:CORRection:ENR:CALibration:TABLE:SERial:DATA <sn>

(Read-Write) Sets and returns the serial number of noise source. This is for informational purposes only to identify the specific noise source for which the data pertains. [Learn more about ENR files.](#)

Parameters

<sn> Serial number of the noise source for which the data applies, enclosed in quotes.

Examples

```
SENS:CORR:ENR:CAL:TABL:SER:DATA "ABCD1234"  
sense:correction:enr:calibration:table:serial:data "ABCD1234"
```

Query Syntax SENSE:CORRection:ENR:CALibration:TABLE:SERial:DATA?

Return Type String

Default Not Applicable

SENSe<ch>:CORRection:GCSetup:POWER <num>

(Read-Write) Set and read the power level at which to perform the Source Power Cal portion of a Gain Compression (Opt 086) Calibration. [Learn more about this setting.](#)

Parameters

<num> Power level in dB. Choose a value from +30 to (-30).

Examples

```
SENS:CORR:GCS:POW 0  
sense:correction:gcsetup:power 5
```

Query Syntax SENSE:CORRection:GCSetup:POWER?

Return Type Numeric

Default 0

SENSe<ch>:CORRection:GCSetup:SENSor:CKIT <string>

(Read-Write) Set and read the cal kit to be used for calibrating at the port 1 reference plane when the power sensor connector is different from the DUT port 1. [Learn more.](#)

Parameters

<string> Cal Kit. Use [SENS:CORR:COLL:GUID:CKIT:PORT1:CAT?](#) to return a list of valid cal kits.

Examples

```
SENS:CORR:GCS:SENS:CKIT "85052B"
```

Query Syntax SENSe:CORRection:GCSetup:SENSor:CKIT?

Return Type String

Default Not Applicable

SENSe<ch>:CORRection:GCSetup:SENSor:CONNector<string>

(Read-Write) Set and read the power sensor connector type which is used to perform the Source Power Cal portion of a Gain Compression Calibration. [Learn more.](#)

Parameters

<string> Power sensor connector type. Use [SENS:CORR:COLL:GUID:CONN:CAT?](#) to return a list of valid connector types.

Select "Ignored" to NOT compensate for the adapter.

Examples

```
SENS:CORR:GCS:SENS:CKIT "3.5 mm (50) male"
```

Query Syntax SENSe:CORRection:GCSetup:SENSor:CONNector?

Return Type String

Default Not Applicable

SENSe:CORRection:IMPedance:INPut:MAGNitude <num>

(Read-Write) Sets and returns the system impedance value for the analyzer.

Parameters

<num> System Impedance value in ohms. Choose any number between 0 and 1000 ohms.

Examples

```
SENS:CORR:IMP:INP:MAGN 75  
sense:correction:impedance:input:magnitude 50.5
```

Query Syntax SENSE:CORRection:IMPedance:INPut:MAGNitude?

Return Type Numeric

Default 50

SENSe<ch>:CORRection:INTerpolate[:STATe] <ON | OFF>

(Read-Write) Turns correction interpolation ON or OFF.

Note: Before using this command you must select a measurement using [CALC:PAR:SEL](#). You can select one measurement for each channel.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<ON | OFF> **ON** (or 1) - turns interpolation ON.
OFF (or 0) - turns interpolation OFF.

Examples

```
SENS:CORR:INT ON  
sense2:correction:interpolate:state off
```

Query Syntax SENSe<cnum>:CORRection:INTerpolate[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

SENSe<ch>:CORRection:ISOLation[:STATe] <ON | OFF> OBSOLETE

This command no longer works beginning in the PNA 5.2 release. The set and query of this command will NOT return an error.

To perform isolation as part of an unguided calibration, you must explicitly measure the isolation standard using [SENS:CORR:COLL:ACQ Stan5](#).

To measure isolation as part of an ECal, use [SENS:CORR:COLL:ISOL:ECAL](#).

(Read-Write) Turns isolation cal ON or OFF during Full 2-port calibration. If this command is not sent, the default state is to **disable** Isolation.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <ON | OFF> **ON** (or 1) - turns isolation ON.
- OFF** (or 0) - turns isolation OFF.

Examples

```
SENS:CORR:ISOL ON  
sense2:correction:isolation:state off
```

Query Syntax SENSE<cnum>:CORRection:ISOLation[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF - (Isolation disabled)

SENSe<ch>:CORRection:METhods:MATCh <bool>

(Read-Write) Turns match-correction ON or OFF. Use this command AFTER performing an Guided Power Cal. [Learn more](#).

Parameters

- <ch> Channel number on which Guided Power Cal was performed. If unspecified, value is set to 1
- <bool> **ON** (or 1) - Turns match-correction ON
- OFF** (or 0) - Turns match-correction OFF.

Examples

```
SENS:CORR:METH:MATC 0  
sense2:correction:methods:match off
```

Query Syntax SENSE<cnum>:CORRection:METhods:MATCh?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

SENSe:CORRection:PREFeRence:CALibration[:FOM]:RANGe <char>

(Read-Write) Specifies the FOM frequency range to use when performing calibration.

Parameters

<char> Choose from:

PRIMary - Used for calibrating at the mmWave frequencies when NOT using a test set. [Learn more.](#)

AUTO - All other calibration situations.

Examples

```
SENS:CORR:REF:CAL:RANG PRIM
sense:correction:preference:calibration:fom:range auto
```

Query Syntax SENSe:CORRection:PREFeRence:CALibration[:FOM]:RANGe?

Return Type Character

Default AUTO

SENSe:CORRection:PREFeRence:CSET:SAVE <char>

Important Notes:

- This command replaces [SENS:CORR:REF:CSET:SAVU](#)
- With 6.0 we implemented a change that defaults to saving completed calibrations to Cal Registers instead of User Cal Sets. To revert to the old behavior, send this command with the USER argument.

(Read-Write) Specifies the default manner in which calibrations that are performed using SCPI or COM are to be stored. Cal data is ALWAYS stored to the channel Cal Register regardless of this setting.

This setting survives instrument preset and reboot. It remains until changed by another execution of this command.

Note: Cal Set arguments used with commands such as [SENS:CORR:COLL:GUID:INIT](#), [SENS:CORR:COLL:GUID:SAVE](#) and [SENS:CORR:COLL:GUID:SAVE:CSET](#) will override of any of these default preference settings.

Learn about [Cal Registers and User Cal Sets](#).

Parameters

<char> **CALRegister** - Each Cal is saved ONLY to the channel Cal Register. If the error terms from a new Cal can co-exist with those in the Cal Register, they are

appended.

USER - Each Cal is saved to its own new User Cal Set file. The Cal Set name is automatically generated. To change the name, send [SENS:CORR:CSET:NAME](#) after the cal is complete. This reverts to pre-6.0 behavior.

REUSE - The cal is saved to the Cal Set that is currently selected on the specified channel, which could be the channel Cal Register. If the channel does not yet have a selected Cal Set, the cal will be saved to a new User Cal Set with an automatically-generated name. If the error terms from a new Cal can co-exist with those in the Cal Set, they are appended.

Examples

```
SENS:CORR:PREF:CSET:SAVE USER
sense:correction:preference:cset:save reuse
```

Query Syntax SENSE:CORRection:PREFeRence:CSET:SAVE?

Return Type Character

Default CALRegister

SENSe:CORRection:PREFeRence:CSET:SAVUser <bool> **Superseded**

This command is replace with [SENS:CORR:PREF:CSET:SAVE](#)

NOTE: With 6.0 we implemented a change that defaults to saving completed calibrations to Cal Registers instead of User Cal Sets. To revert to the old behavior, send this command as ON (1). For UI and COM use, this can be done from the [GPIB console](#).

(Read-Write) Specifies whether cal data is automatically saved to a User Cal Set file after performing a SCPI calibration. Cal data is always saved to a Cal Register regardless of this setting.

This setting survives instrument preset and reboot. It remains until changed by another execution of this command.

Learn about [Cal Registers and User Cal Sets](#).

Parameters

<bool> **ON** or **1** - Cal is automatically saved to a User Cal Set file when performing a SCPI calibration. The Cal Set name is automatically generated. To change the name, send [SENS:CORR:CSET:NAME](#) after the cal is complete. Reverts to pre-6.0 behavior.

OFF or **0** - Cal is NOT automatically saved to a User Cal Set. To save a calibration to a User Cal Set, use [SENS:CORR:COLL:GUID:INIT](#).

Examples

```
SENS:CORR:PREF:CSET:SAVU 1
sense:correction:preference:cset:savuser 0
```

Query Syntax SENSE:CORRection:PREFErence:CSET:SAVUser?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF (0)

SENSe:CORRection:PREFErence:ECAL:ORientation[:STATe] <ON|OFF>

(Read-Write) Specifies whether or not the PNA should perform orientation of the ECal module during calibration. Orientation is a technique by which the PNA automatically determines which ports of the module are connected to which ports of the PNA. Orientation begins to fail at very low power levels or if there is much attenuation in the path between the PNA and the ECal module. If orientation is turned OFF, the [SENS:CORR:PREF:ECAL:PMAP](#) command must be used to specify the port connections before performing a cal.

Note: For 3-port or 4-port measurements, when orientation is OFF, you are not allowed to specify how the ECAL module is connected. Instead, the PNA determines the orientation. Use [SENS:CORR:COLL:GUID:DESC?](#) to query the orientation. The PNA does not verify that you made the connection properly.

This setting remains until the PNA is restarted or this command is sent again.

Parameters

<bool> ECAL orientation state. Choose from:

ON or **1** - PNA performs orientation of the ECal module.

OFF or **0** - PNA does NOT performs orientation of the ECal module.

Examples

```
SENS:CORR:PREF:ECAL:ORI OFF
sense:correction:preference:ecal:orientation:state on
```

Query Syntax SENSE:CORRection:PREFErence:ECAL:ORientation[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON (1)

SENSe:CORRection:PREFErence:ECAL:PMAP <module>,<string>

(Read-Write) When ECal module orientation is turned OFF ([SENS:CORR:PREF:ECAL:ORI](#)), this command specifies the port mapping (which ports of the module are connected to which ports of the PNA) prior to performing ECal calibrations.

This setting remains until the PNA is restarted or this command is sent again.

Parameters

<module> Specifies which ECal module this port map is being applied to. Choose from:

ECAL1

.through.

ECAL50

<string> Format this parameter in the following manner:

Aw,Bx,Cy,Dz

where

- A, B, C, and D are literal ports on the ECal module
- w,x,y, and z are substituted for PNA port numbers to which the ECal module port is connected.

Ports of the module which are not used are omitted from the string.

For example, on a 4-port ECal module with

port A connected to PNA port 2

port B connected to PNA port 3

port C not connected

port D connected to PNA port 1

the string would be: A2,B3,D1

If either the receive port or source port (or load port for 2-port cal) of the CALC:PAR:SElected measurement is not in this string and orientation is OFF, an attempt to perform an ECal calibration will fail.

Examples

```
SENS:CORR:PREF:ECAL:PMAP ECAL2, 'A1,B2'  
sense:correction:preference:ecal:pmap ecal3, 'a2,b1,c3'
```

Query Syntax SENSE:CORRection:PREFErence:ECAL:PMAP? <module>

Return Type String

Default Null string ()

SENSe:CORRection:PREFereNce:SIMCal <bool> **Obsolete**

This command is no longer supported. [Learn more about old and new behaviors.](#)

(Read-Write) Sets and returns a preference for the Unguided Cal behavior described below. This setting persists until it is changed.

This preference can also be set ON by executing the script on the PNA at C:/Program Files/Agilent/Network Analyzer/System/wincal32.reg.

Parameters

<bool> Boolean - Choose from:

0 - OFF - Reverts to new (preferred) behavior. An error is returned if standard data is not acquired before sending [SENS:CORR:COLL:SAVE](#).

1 - ON - (WinCal compatible) Prevents [SENS:CORR:COLL:SAVE](#) from failing when standard data has not, and will not, be acquired.

Examples

```
SENSe:CORR:PREF:SIMC 0
```

```
sense:correction:preference:simcal 1
```

Query Syntax SENSe:CORRection:PREFereNce:SIMCal?

Return Type Boolean

Default 0

SENSe:CORRection:PREFereNce:TRIG:FREE <char>, <bool>

(Read-Write) Sets and returns the preference for the trigger behavior during a calibration. This setting persists until it is changed.

Note: If [TRIGger:SOURce](#) = Manual, during a calibration the PNA ALWAYS switches to Internal for one trigger, then back to Manual, regardless of this preference command.

Parameters

<char> Character - Calibration type. Choose from:

GUIDed - preference setting pertains to a Guided calibration.

UNGUIDed - preference setting pertains to an Unguided calibration.

<bool> Boolean - Choose from:

0 - OFF - The trigger behavior during the specified calibration type DOES respect the setting of the [TRIGger:SOURce](#) command. For example, when

Trigger source = External, the single trigger method will wait for the External trigger signal and then allow only one sweep.

1 - ON - (Pre-6.0 behavior) The trigger behavior during the specified calibration type does NOT respect the setting of the [TRIGger:SOURce](#) command. For example, when Trigger source = External, during calibration the PNA switches to Internal sweep, responds to one trigger signal to measure the standard, then switches back to External.

Examples

```
SENS:CORR:PREF:TRIG:FREE GUID,1  
sense:correction:preference:trig:free unguided,0
```

Query Syntax SENSE:CORRection:PREFeRence:TRIG:FREE? <char>

Return Type Boolean

Default OFF for both calibration types.

SENSe<cnum>:CORRection:RPOWER:OFFSet[:AMPLitude] <num>

(Read-Write) Adjusts a receiver power cal to account for components or adapters that are added between the source port and receiver while performing this cal. For more information, see [Receiver Cal.](#)

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<num> Offset Value in dB. Specify loss as a negative number; and gain as a positive number. Choose a number between -200 and 200.

Examples

```
SENS:CORR:RPOW:OFFS .5  
sense2:correction:rpower:offset:amplitude .-5
```

Query Syntax SENSe<cnum>:CORRection:RPOWER:OFFSet[:AMPLitude]?

Return Type Numeric

Default 0

SENSe<cnum>:CORRection:RVELocity:COAX <num>

(Read-Write) Sets the velocity factor to be used with Electrical Delay and Port Extensions.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<num> Velocity factor. Choose a number between **0** and **10**

(.66 polyethylene dielectric; .7 PTFE dielectric)

Examples

```
SENS:CORR:RVEL:COAX .66  
sense2:correction:rvelocity:coax .70
```

Query Syntax SENSE<cnum>:CORRection:RVELocity:COAX?

Return Type Numeric

Default 1

SENSe<cnum>:CORRection:SFORward[:STATe] <boolean>

(Read-Write) Sets the direction a calibration will be performed when only one set of standards is used.

Use [SENSe:CORRection:TSTandards\[:STATe\]](#) **OFF** to specify that only one set of standards will be used.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<boolean> **ON (1)** - FORWARD direction of a 2-port calibration will be performed

OFF (0) - REVERSE direction of a 2-port calibration will be performed

Examples

```
SENS:CORR:SFOR 1  
sense2:correction:sforward:state 0  
  
See an example using this command
```

Query Syntax SENSe<cnum>:CORRection:SFORward[:STATe]?

Return Type Boolean

Default ON

SENSe<cnum>:CORRection[:STATe] <ON | OFF>

(Read-Write) Turns error correction ON and OFF for the specified channel.

Note: Before using this command you must select a measurement using [CALC:PAR:SEL](#). You can select one measurement for each channel.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <ON | OFF> **ON** (or 1) - correction is applied to the channel.
- OFF** (or 0) - correction is NOT applied to the channel.

Examples

```
SENS:CORR ON  
sense2:correction:state off
```

Query Syntax SENSE<cnum>:CORRection[:STATe]?

To query the error correction state for a measurement, use [CALC:CORR:STATe?](#)

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

SENSe<cnum>:CORRection:TCOLd:USER:VALue <num>

(Read-Write) Sets and returns the temperature of the noise source connector. Learn more about [Noise Figure Calibration](#).

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1.
- <num> Noise source temperature in Kelvin.

Examples

```
SENS:CORR:TCOL:USER:VAL 295  
  
sense2:correction:tcold:user:value 298  
  
See an example using this command
```

Query Syntax SENSe<cnum>:CORRection:TCOLd:USER:VALue?

Return Type Numeric

Default Not Applicable

SENSe<cnum>:CORRection:TSTandards[:STATe] <boolean>

(Read-Write) Specifies the acquisition of calibration data using ONE or TWO sets of standards.

Parameters

- <num> Any existing channel number. If unspecified, value is set to 1.
- <boolean> **ON (1)** - TWO identical sets of standards will be used to simultaneously calibrate two ports (for both Forward and Reverse parameters).
- OFF (0)**- ONE set of standards will be used to perform a full 2-port calibration, one port at a time.

When specifying ON (use two sets), the [SENS:CORR:COLL:ACQUIRE](#) command uses the same standard index for each calibration class. To specify the calibration standard gender for each port, you must first ensure that the order of calibration class accurately reflects the configuration of your DUT.

For example, for a DUT with a male connector on port 1 and a female connector on port 2, order the devices within the S11 classes (A, B, and C) such that the MALE standards are first in the list. Then order the S22 classes specifying the FEMALE standards as the first in the list.

Examples

```
SENS:CORR:TST 1
sense2:correction:tstandard:state 0
```

See an [example](#) using this command

Query Syntax SENSE<num>:CORRection:TSTandards[:STATe]?

Return Type Boolean

Default ON

SENSe:CORRection:TYPE:CATalog? <char>

(Read-Write) Lists the Cal Types in the PNA by either GUID or registered name. [Learn more about applying Cal Type using SCPI.](#)

Note: Before using this command you must select a measurement using [CALC:PAR:SEL](#). You can select one measurement for each channel.

Parameters

<char> Specifies the type of list. Choose from:

GUID - the registered GUID of the Cal Type

NAME - the registered name of the Cal Type

Examples

```
SENS:CORR:TYPE:CAT? GUID
```

Query Syntax SENSE<cnum>:CORREction:TYPE:CATalog? <char>

Return Type Comma-separated string

Default Not Applicable

Last modified:

16-Jan-2014	Obsolete WinCal Pref command
16-Jan-2014	Fixed link at ENR:CAL:Table:Data
16-Jan-2014	Added note to Obsolete Isolation command
	Reordered list of commands
16-Jan-2014	Removed errant link to Window:STATe
	Updated ACQUIRE command with ECal# and char
	Fixed TCOLd example
	Added Cal method Sparsolt3
16-Jan-2014	Edited Pref:Save
16-Jan-2014	Edited TRL limitations
	Added SENS:CORR:METH:MATC
16-Jan-2014	Updated confidence check to 12 User chars
16-Jan-2014	Added Noise commands and Normalization method. (9.1)
16-Jan-2014	Added Cal Pref FOM
16-Jan-2014	Added Acq, sync
16-Jan-2014	Added 'if meas deleted' to Swe:Chan
16-Jan-2014	Added Noise TCOLd

16-Jan-2014	Added missing <cnun> arguments
January 16, 2014	Added ENR commands
January 16, 2014	Add ECal isolation commands
Oct 30, 2006	Modified SavUser command

Sense:Correction:CKIT Commands

Manages the list of cal kits that are installed in the PNA.

SENSe:CORR:CKIT

[CLEar](#)

[COUNT?](#)

ECAL

| [CHARacterize More commands](#)

| [CLIST?](#)

| **DMEMory**

| [CLEar](#)

| [IMPort](#)

| [EXPort](#)

| [INFormation](#)

| **KNAMe**

| [INFormation](#)

| [LIST?](#)

| [ORient?](#)

| **PATH**

| [COUNT?](#)

| [DATA?](#)

EXPort

IMPort

INITialize

LOAD

- Click on a [blue](#) keyword to view the command details.
- Red is a superseded command

- New [See Calibrating the PNA Using SCPI](#)
 - [Learn about Modifying Cal Kits](#)
 - [Synchronizing the PNA and Controller](#)
 - [SCPI Command Tree](#)
-

SENSe:CORRection:CKIT:CLEar[:IMMediate] [ckit]

(Write-only) Deletes installed cal kits.

Parameters

[ckit] Optional String. Cal Kit to delete. If not specified, all PNA Cal kits are deleted, including custom kits.

Examples

```
SENS:CORR:CKIT:CLE
```

```
sense:correction:ckit:clear:immediate "85052B"
```

Query Syntax Not Applicable

Default Not Applicable

SENSe:CORRection:CKIT:COUNT?

(Read-only) Returns the number of installed cal kits.

Examples

```
SENS:CORR:CKIT:COUNT?
```

Query Syntax SENS:CORR:CKIT:COUNT?

Return Type Numeric

Default Not Applicable

SENSe:CORRection:CKIT:ECAL<mod>:CLISt?

(Read-only) Returns a list of characterizations stored in the specified ECal module.

Parameters

<mod> ECal module from which to read user characterization numbers. Choose from 1 to 50. If unspecified, value is set to 1.

Examples

```
Module 1 contains User Characterizations 1 and 3.  
  
SENSe:CORRection:CKIT:ECAL:CLIST?  
  
'Returns the following (0 always indicates the factory  
characterization):  
  
0,1,3
```

Return Type Numeric list, separated by commas.

Default Not Applicable

SENSe:CORRection:CKIT:ECAL:DMEMory:CLEar <kitName>

(Write-only) Deletes user characterizations from PNA disk memory.

Parameters

<kitName> Optional String argument. ECal Model, User Characterization name + " ECal", and serial number of the ECal module, separated by spaces. See examples below.

If unspecified, ALL User Characterizations that are stored in PNA disk memory are deleted.

Examples

```
'These examples all use "MyUserChar" as the User  
characterization name.  
  
'The "My User Char" characterization is deleted from disk  
memory.  
  
SENS:CORR:CKIT:ECAL:DMEM:CLE "N4433A MyUserChar ECal 00001"  
  
'All User characterizations are deleted from disk memory.  
  
SENS:CORR:CKIT:ECAL:DMEM:CLE
```

Query Syntax Not Applicable

Default Not Applicable

SENSe:CORRection:CKIT:ECAL:DMEMory:IMPort <file>

(Write-only) After the PNA disk memory is [Exported](#) to a file, use this command to Import the file into PNA disk memory, which allows the User Characterization to be used with the PNA and ECal module.

Note: An ECal confidence check can NOT be performed remotely from User Characterizations that are stored on the PNA disk.

Parameters

<file> String. Full path and file name of file that was exported.

Examples

```
SENS:CORR:CKIT:ECAL:DMEM:IMP "c:\users\public\network  
analyzer\ECal User Characterizations/myDiskUserChar.euc"
```

Query Syntax Not Applicable

Default Not Applicable

SENSe:CORRection:CKIT:ECAL:EXPort <kit>[,<file>][,<NewName>]]

(Write-only) Saves an existing ECal characterization to a file. Use this command to archive the user characterization or to move the characterization to a different PNA for use with the specified ECal module. After exporting the user characterization, use [SENS:CORR:CKIT:ECAL:DMEM:IMP](#) to make the user characterization available for use.

Parameters

<kit> String. Not case sensitive. ECal Model, User char name + " ECal", and serial number of the ECal module used for the characterization, separated by spaces. See examples below.

If the model and serial number of the module is not found, an error is returned.

[<file>] Optional String argument. Path and filename of the user characterization. If not specified, the file is saved using characterization name + ".euc". If the path is not specified, it is stored in C:/Program Files/Agilent/Network Analyzer/ECal User Characterizations/. The extension ".euc" is appended if one is not specified.

[<NewName>] Optional String argument. This allows you to change the name for the User Characterization. When specified, the new name is saved in the file with the characterization. If unspecified, the existing user characterization name is saved.

Note: If this argument is specified, the second argument (<file>) must also be specified.

Examples

```
'These examples all use "MyUserChar" as the User
```

```
characterization name.
```

```
'All parameters specified
```

```
SENS:CORR:CKIT:ECAL:EXP "N4433A MyUserChar ECal  
00001", "myUserChar.euc", "NewUserChar"
```

```
'First two parameters are specified
```

```
sense:correction:ckit:ecal:export "N4691B MyUserChar ECal  
00500", "myUserChar.euc"
```

```
'Only first parameter is specified
```

```
SENS:CORR:CKIT:ECAL:EXP "N4433A MyUserChar ECal 00001"
```

Query Syntax Not Applicable

Default Not Applicable

SENSe:CORRection:CKIT:ECAL<mod>:INFormation? [<char>]

(Read-only) Reads the identification and characterization information from the specified ECal module.

Note: To read user-characterization information that is stored in PNA disk memory, then use [SENS:CORR:CKIT:ECAL:KNAM:INF?](#)

Parameters

<mod> ECal module from which to read characterizations. Choose from 1 through 50. If unspecified, value is set to 1.

Do NOT assume the <mod> number is the order in which ECal modules were connected.

Use [SENS:CORR:CKIT:ECAL:LIST?](#) to read a list of <mod> numbers of currently-attached ECal modules.

<char> Optional argument. Specifies which characterization to read information from. If not specified, value is set to CHAR0.

Choose from:

- CHAR0 Factory characterization (data that was stored in the ECal module by Agilent)
- CHAR1 User characterization #1
- CHAR2 User characterization #2

- - through -
- CHAR12 User characterization #12

Examples

```
SENS:CORR:CKIT:ECAL2:INformation? char5
```

```
'Example return string:
```

```
"ModelNumber: 85092-60007, SerialNumber: 01386, ConnectorType:
N5FN5F, PortAConnector: Type N (50) female, PortBConnector: Type
N (50) female, MinFreq: 30000, MaxFreq: 9100000000,
NumberOfPoints: 250, Calibrated: July 4 2002"
```

Return Type Character

Default Not Applicable

SENSe:CORRection:CKIT:ECAL:KNAME:INformation? <kitName>

(Read-only) Reads the identification and characterization information from the specified ECal module or PNA disk memory.

[Learn more about User Characterization in PNA Disk Memory.](#)

Parameters

<kitName> String. ECal model and characterization to read information from, enclosed in quotes, in the following format:

```
<model> <name> ECal <serial number>
```

Where:

<model>: Always required

<name>:

- For the factory characterization, do not specify.
- For a user-characterization stored in the module, use **User <n>** in the string, where <n> is the user-characterization number. Not case sensitive. Separate User and <n> with a space.
- For a user-characterization stored in PNA disk memory, use <charName> from [SENS:CORR:CKIT:ECAL:CHAR:DMEM:SAVE <charName>](#)

ECal - not case sensitive

<serial number>: Optional. Include when two or more ECal modules with same model number are attached to the PNA,

Each item is separated with a space.

Examples

```
'For a factory characterization in module memory:
SENS:CORR:CKIT:ECAL:KNAM:INF? "N4433A ECal"

'For user characterization in module memory with optional serial
number:
SENS:CORR:CKIT:ECAL:KNAM:INF? "N4433A User 1 ECal 00028"

'For user characterization "foo" in disk memory:
SENS:CORR:CKIT:ECAL:KNAM:INF? "N4433A foo ECal 00028"

'Example return string:

"ModelNumber: N4433A, SerialNumber: 00028, ConnectorType:
N5FN5F, PortAConnector: Type N (50) female, PortBConnector: Type
N (50) female, MinFreq: 30000, MaxFreq: 9100000000,
NumberOfPoints: 250, Calibrated: July 4 2002"
```

Return Type String

Default Not Applicable

SENSe:CORRection:CKIT:ECAL:LIST?

(Read-only) Returns a list of index numbers for ECal modules that are currently attached to the PNA. Use these numbers (called <mod> in PNAHelp) to refer to the ECal module using SCPI commands.

Examples

```
SENS:CORR:CKIT:ECAL:LIST?

'If 2 modules are attached to the PNA
'then the returned list will be:

+1,+2

'If NO modules are attached to the PNA
'then the returned list will be:

+0

See example program using this command.
```

Return Type Numeric list, separated by commas.

Default Not Applicable

SENSe<ch>:CORRection:CKIT:ECAL<n>:ORlent? <pnaPort>[,<charN>]

(Read-only) Returns the ECal port that is connected to the specified PNA port. A calibration does not have to be in process.

<ch> Channel number that contains the frequency range to be calibrated.

<n> ECal module number. Choose from 1 through 50.

If unspecified (only one ECal module is connected to the USB), <n> is set to 1.

If two or more modules are connected, use [SENS:CORR:CKIT:ECAL:LIST?](#) to determine how many, and [SENS:CORR:CKIT:ECAL:INF?](#) to verify their identities.

<pnaPort> PNA port number.

<charN> Optional argument. If unspecified, factory data (CHAR0) is used. User Characterization number that matches the physical adapters/fixtures that are on the ECal module. This aids in determining the orientation of the ECal module.

Choose from:

- **CHAR0** Factory characterization (data that was stored in the ECal module by Agilent)
- **CHAR1** User characterization #1
- **CHAR2** User characterization #2

and so forth up to:

- **CHAR12** User characterization #12

Beginning with A.08.33, up to 12 User Characterizations can be stored in a single ECal module. Previous releases allowed up to 5. [Learn more.](#)

Examples

```
SENS1:CORR:CKIT:ECAL1:ORI? 2
```

```
sense2:correction:ckit,ecal1:orient? 2, char2
```

Return Type The returned ECal port number is a 1-based number: 1 = Port A, 2 = Port B, 3 = Port C, 4 = Port D.

Zero (0) is returned when the auto-orientation routine is unable to resolve the orientation.

Default Not Applicable

SENSe:CORRection:CKIT:ECAL<n>:PATH:COUNT? <path>

(Read-only) Returns the number of unique states that exist for the specified path name on the selected ECal module.

This command performs exactly the same function as [CONT:ECAL:MOD:PATH:COUNT?](#)

Use the [CONT:ECAL:MOD:PATH:STAT](#) command to set the module into one of those states.

Use [SENS:CORR:CKIT:ECAL:PATH:DATA?](#) to read the data for a state.

Parameters

<n> USB number of the ECal module. Choose from 1 to 50.

If unspecified (only one ECal module is connected to the USB), <n> is set to 1. If two or more modules are connected, use [SENS:CORR:CKIT:ECAL:LIST?](#) to determine how many, and [SENS:CORR:CKIT:ECAL:INF?](#) to verify their identities.

<path> Name of the path for which to read number of states. Choose from:

Reflection paths

- **A**
- **B**
- **C** (4-port modules)
- **D** (4-port modules)

Transmission paths

- **AB**
- **AC** (4-port modules)
- **AD** (4-port modules)
- **BC** (4-port modules)
- **BD** (4-port modules)
- **CD** (4-port modules)

Examples

```
CONT:ECAL:MOD:PATH:COUNT?  
control:ecal:module2:path:count?
```

Return Type Integer

Default Not Applicable

SENSe<ch>:CORRection:CKIT:ECAL<num>:PATH:DATA? <path>, <stateNum>[,<char>]

(Read-only) Returns the data for a state from the memory of the selected ECal module. The returned data is interpolated if necessary to have the same stimulus values as the specified channel <ch>.

- For a reflection path state, the data is reflection S-parameter data. The number of values equals the number of stimulus points on the channel multiplied by 2 (because they are complex numbers).
- For a transmission path state, the data is all 4 S-parameters of the state. The number of values returned is 4 times that of a reflection state.

The data is returned in the same format as [CALC:DATA:SNP?](#)

Note: This command returns SNP data without header information, and in columns, not in rows as .SnP files. This means that the data returned from this command sends all frequency data, then all Sx1 magnitude or real data, then all Sx1 phase or imaginary data, and so forth.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
- <num> Optional argument. USB number of the ECal module. Choose from 1 through 50.

If unspecified (only one ECal module is connected to the USB), <num> is set to 1. If two or more modules are connected, use [SENS:CORR:CKIT:ECAL:LIST?](#) to determine how many, and [SENS:CORR:CKIT:ECAL:INF?](#) to verify their identities.

- <path> Name of the path for which to read number of states. Choose from:

Reflection paths

- **A**
- **B**
- **C** (4-port modules)
- **D** (4-port modules)

Transmission paths

- **AB**
- **AC** (4-port modules)
- **AD** (4-port modules)

- **BC** (4-port modules)
- **BD** (4-port modules)
- **CD** (4-port modules)

<stateNum> Number of the state to set. Refer to the following table to associate the <stateNum> with a state in your ECal module.

In addition, [CONT:ECAL:MOD:PATH:COUNt?](#) returns the number of states in the specified ECal module.

<stateNum>	N4432A and N4433A States	N4431A States	N469x States**	8509x States
One-Port Reflection States				
1	Open	Open	Impedance 1	Open
2	Short	Short	Impedance 2	Short
3	Impedance 1	Impedance 1	Impedance 3	Impedance 1
4	Impedance 2	Impedance 2	Impedance 4	Impedance 2
5			Impedance 5	
6			Impedance 6	
7			Impedance 7	
Two-Port Transmission States				
1	Thru	Thru	Thru	Thru
2	Confidence	Confidence	Confidence	Confidence

** The following modules have only FOUR Impedance states (1, 2, 3, 4):
N4690B ,N4691B ,N4692A ,N4696B.

<char> Optional argument. Specifies which characterization within the ECal module to read information from. If not specified, value is set to CHAR0.

Choose from:

- **CHAR0** Factory characterization (data that was stored in the ECal module by Agilent)
- **CHAR1** User characterization #1
- **CHAR2** User characterization #2

and so forth up to:

- **CHAR12** User characterization #12

Examples `SENS:CORR:CKIT:ECAL1:PATH:DATA? A,1`

Return Type S1P or S2P

Default Not Applicable

SENSe:CORRection:CKIT:EXPort <kit>[,<file>]

(Write-only) Saves an existing cal kit definitions to a file. Use this command to archive or move a user-defined or modified cal kit to a different PNA. After exporting the cal kit, use [SENSe:CORR:CKIT:IMPort](#) to make the cal kit available for use on the PNA. This command provides the same behavior as the Installed Kits - Save As button on the [Edit PNA Cal Kits](#) dialog.

Parameters

- <kit> String. Not case sensitive. Name of the cal kit to export, as seen in the Cal Kits field of the [Select DUT Connectors and Cal Kits](#) dialog of a SMART Cal.
- <file> Optional String argument. Path and filename to where the Cal Kit file is to be saved. If not specified, the file is saved using <kit> + ".ckt". If the path is not specified, it is stored in C:/Program Files/Agilent/Network Analyzer/PNACalKits/User.

Examples

```
'File unspecified
SENS:CORR:CKIT:EXP "MyCalKit"

'Both parameters are specified
sense:correction:ckit:export "MyCalKit","C:/myBackupCalKit.ckt"
```

Query Syntax Not Applicable

Default Not Applicable

SENSe:CORRection:CKIT:IMPort <string>

(Write-only) Imports the specified cal kit (.ckt file) and appends the imported kit to the end of the list of kits.

Note: Although there is no limit to the number of cal kits that can be imported, during an [Unguided cal](#), you can access ONLY mechanical cal kits #1 through #95.

Parameters

<string> Path and cal kit name.

Examples

```
SENSe:CORRection:CKIT:IMPort "c:\users\public\network  
analyzer\documents\85033D.ckt"
```

Query Syntax Not Applicable

Default Not Applicable

SENSe:CORRection:CKIT:INITialize[:IMMediate] [ckit]

(Write-only) Restores default factory installed cal kits. This command also selects kit number 1, as you would using [SENS:CORR:COLL:CKIT:SEL 1](#). Therefore, if you intend to work with a Cal Kit remotely, select the Cal Kit **AFTER** sending this command.

Note: This command can also delete all existing User-defined Cal Kits. However, if saved using Save As, these kits can be restored in the same manner as after a PNA firmware upgrade. [Learn more about saving modified Cal Kits.](#)

Parameters

[ckit] Optional String. Cal Kit to restore. If not specified, all PNA factory Cal kits are restored.

Examples

```
SENS:CORR:CKIT:INITialize  
sense:correction:ckit:initialize:immediate "85052B"
```

Query Syntax Not Applicable

Default Not Applicable

SENSe:CORRection:CKIT:LOAD <string>

(Write-only) Loads the specified collection of cal kits from a .wks file. You can make your own collection of cal kits from the [Advanced Modify Cal Kit](#) menu.

Parameters

<string> Path and file name of the cal kit collection.

Examples

```
sense:correction:ckit:load "C:/Program Files/Agilent/Network Analyzer/PnaCalKits/factory/wMyCalKits.wks"
```

Query Syntax Not Applicable

Default Not Applicable

Last modified:

- 17-Jan-2014 Removed superseded from INF? command
- 5-Nov-2013 Edits to SENS:CORR:CKIT:INIT
- 16-Aug-2013 Updated ECal nums
- 23-Jan-2013 Fixed several string examples
- 12-Oct-2011 Edit Import command
- 3-Aug-2011 Edit ECal list
- 11-Apr-2011 Edited Orient? command
- 15-Jun-2010 Updated for 12 User Chars
- 19-May-2009 Added [ckit] argument to clear and init
- 24-Mar-2009 Edited Path SNP
- 6-Mar-2009 Added ECal orient?
- 31-Oct-2008 Added Characterizations (8.33)
- 16-Jun-2008 Added CKIT:INIT note
- 10/16/06 Modified Ecal:Data to include <ch>

Sense:Correction:Collect:Ckit Commands

Use to change the definitions of calibration kit standards.

SENSe:CORRection:COLLect:CKIT:

| **CATalog?**

| **CONNector**

| **ADD**

| **CATalog?**

| **DELeTe**

| **FNAME**

| **SNAME**

| **DESCription**

| **INFormation?**

| **NAME**

| **OLAB**

| **OLISt?**

| **ORDer**

| **PORT[:SElect]**

| **RESet**

| **SElect**

| **STANdard**

| **CO, C1, C2, C3**

| **CHARacter**

| **DELay**

| **FMAX**

| **FMIN**

| **IMPedance**

| **LO, L1, L2, L3**

| [LABel](#)
| [LOSS](#)
| [REMove](#)
| [SDEscription](#)
| [\[SElect\]](#)
| [TYPE](#)
| [TZReal](#)
| [TZImag](#)
| [TRLoption](#)
| [IMPedance](#)
| [LRLChar](#)
| [RPLane](#)

Click on a [blue](#) keyword to view the command details.

[Red](#) keywords are superseded commands.

Most of these commands act on the currently selected standard from the currently selected calibration kit.

- To select a Calibration kit, use [SENS:CORR:COLL:CKIT:SEL](#).
- To select a Calibration standard, use [SENS:CORR:COLL:CKIT:STAN:SEL](#)
- See an **example** program that [CREATES a New Cal Kit](#)
- See an **example** program that [MODIFIES an Existing Cal Kit](#)
- [Learn about Modifying Cal Kits](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

Note: You should provide data for every definition field - for every standard in your calibration kit. If a field is not set, the default value may not be what you expect.

For more information, read [Specifying Calibration Standards and Kits for Agilent Vector Network](#)

SENSe:CORRection:COLLEct:CKIT:CATalog?

(Read-only) Returns the names of the first 50 mechanical cal kits in your PNA that can be used for unguided calibrations.

Examples `SENS:CORR:COLL:CKIT:CAT?`

Return Type A comma-separated string

Default Not Applicable

SENSe:CORRection:COLLEct:CKIT:CONNector:ADD <family>,<start>,<stop>,<z0>,<gender>,<media>,<cutoff>

(Write only) Creates a new connector. The connector is automatically added to the list of available connectors for the currently selected cal kit. If a connector includes both male and female connectors, each connector must be added separately.

Parameters

<family> (String) Name of connector family. Limited to 50 characters.

<start> Start frequency

<stop> Stop frequency

<z0> Characteristic Impedance of the connector in ohms.

<gender> Connector gender. Choose from:

MALE

FEMALE

NONE

<media> Media of the connector. Choose from:

COAX - coaxial

WAVE - waveguide

<cutoff> Cutoff frequency of the connector (waveguide only).

Examples
`SENS:CORR:COLL:CKIT:CONN:ADD "PSC 1.8 mm",0 HZ,999.9
 GHZ,50,FEMALE,COAX,0.0`
`SENS:CORR:COLL:CKIT:CONN:ADD "PSC 1.8 mm",0 HZ,999.9
 GHZ,50,MALE,COAX,0.0`

Query Syntax Not applicable

Default Not Applicable

SENSe:CORRection:COLLect:CKIT:CONNector:CATalog?

(Read-only) Returns a comma-separated list of all connectors defined within the currently selected cal kit. The returned string includes the connector family name followed by the connector gender, if any. Kits may include a primary connector family name and additional connector family names.

Connector family names are case sensitive. A connector family named "PSC 2.4" is different from a connector family named "psc 2.4".

Learn more about [Connector Family Name](#).

Examples

```
SENS:CORR:COLL:CKIT:CONN:CAT?
```

```
'Returned string
```

```
"Type-N (50) male, Type-N (50) female"
```

Default

Not Applicable

SENSe:CORRection:COLLect:CKIT:CONNector:DELeTe

(Write-only) Deletes the primary connector family name from the selected kit. The PNA allows multiple connector families for each kit. If a kit includes multiple connector families, only the first listed (primary) connector family name is deleted.

Once the connector family is deleted, the connector may not be assigned to any new or existing standard within the kit.

The previously defined standards retain their association to the deleted connector name. To reassign standards to a new connector family name, use [SENSe:CORRection:COLLect:CKIT:CONNector:SNAME](#).

Examples

```
SENS:CORR:COLL:CKIT:CONN:DEL
```

Query Syntax

Not Applicable

Default

Not Applicable

SENSe:CORRection:COLLect:CKIT:CONNector:FNAME <name>

(Read-Write) Replaces the primary connector family name from the selected kit with a new connector family name. The connector family name is replaced in all standards in the kit that share that name. The PNA allows multiple connector families for each kit. If a kit includes multiple connector families, only the first listed (primary) connector family name is replaced. Use the query form of this command to return the name of the primary connector family.

Parameters

<name> New connector family name. Limited to 50 characters.

Examples

```
SENS:CORR:COLL:CKIT:CONN:FNAME 'MYPSC35'
```

```
Sense:correction:collect:ckit:connector:name 'My Type N'
```

Query Syntax SENSE:CORRection:COLLect:CKIT:CONNector:FNAME?

Return Type String

Default Not Applicable

SENSe:CORRection:COLLect:CKIT:CONNector:SNAME <family>,<gender>,<port>

(Read-Write) Assigns a family name to the currently selected standard from the currently selected kit. Specify each port of a 2-port standard individually. Use the query form of this command to read the connector family name assigned to the current standard. The name is not assigned unless the connector family name is previously defined within the selected kit.

Parameters

<family> String. Connector family name.

<gender> Connector gender. Choose from:
MALE
FEMALE
NONE

<port> Number of the connector port to be assigned the connector family name. 2-port standards such as a thru line must be assigned separately. It is not relevant which connector is port 1 or port 2.

1 Specifies a 1-port standard or the first port of a 2-port standard.

2 Specifies the second port of a 2-port standard.

Examples

```
SENS:CORR:COLL:CKIT:CONN:SNAME "Type-N (50)",MALE,1
```

Query Syntax SENSe:CORRection:COLLect:CKIT:CONNector:SNAME?

Return Type String

Default Not Applicable

SENSe:CORRection:COLLect:CKIT:DESCRiption <string>

(Read-Write) Modifies the cal kit description field of the selected kit. This description appears in the [Edit PNA Cal Kit dialog box](#).

Parameters

<string> Description of the cal kit. Limited to 50 characters.

Examples

```
SENSe:CORR:COLL:CKIT:DESC "My New CalKit"
```

Query Syntax

```
SENSe:CORRection:COLLect:CKIT:DESCRiption?
```

Return Type

String

Default

Not Applicable

SENSe:CORRection:COLLect:CKIT:INFormation? <module>[,char]

(Read Only) Reads characterization information from an ECal module.

Parameters

<module> Specifies which ECal module to read from. Choose from:

ECAL1

.through.

ECAL50

[char] Optional argument.

Specifies which characterization within the ECal module to read information from. If this argument is not used, the default is **CHAR0**. **CHAR1** through **CHAR5** are for user characterizations that may have been written to the module by the User Characterization feature on the PNA. Choose from:

CHAR0 Factory characterization (data that was stored in the ECal module by Agilent)

CHAR1 User characterization #1

CHAR2 User characterization #2

- through -

CHAR12 User characterization #12

Examples

```
SENS:CORR:COLL:CKIT:INF? ECAL4  
sense:correction:collect:ckit:information? ecal2,char1
```

Example return string:

```
ModelNumber: 85092-60007, SerialNumber: 01386, ConnectorType:  
N5FN5F, PortAConnector: Type N (50) female, PortBConnector: Type  
N (50) female, MinFreq: 30000, MaxFreq: 9100000000,  
NumberOfPoints: 250, Calibrated: July 4 2002
```

Return Type Character

Default Not Applicable

SENSe:CORRection:COLLEct:CKIT:NAME <name>

(Read-Write) Sets a name for the selected calibration kit.

Parameters

<name> Calibration Kit name. Any string name, can include numerics, period, and spaces; any length (although the dialog box display is limited to about 30 characters).

Examples

```
SENS:CORR:COLL:CKIT:NAME 'MYAPC35'  
sense:correction:collect:ckit:name 'mytypen'
```

Query Syntax SENSe:CORRection:COLLEct:CKIT:NAME?

Return Type String

Default Not Applicable

SENSe:CORRection:COLLEct:CKIT:OLABel<class> <name>

(Read-Write) Sets the label for the calibration class designated by <class>. The label is used in the prompts for connecting the calibration standards associated with that <class>.

Parameters

<class> Number of the calibration class. Choose a number between: 1 and 18. The <class> numbers are associated with the following calibration Classes:

	Class	Description
1	S11A	Reflection standard
2	S11B	Reflection standard
3	S11C	Reflection standard
4	S21T	Thru/Delay standard
5	S22A	Reflection standard
6	S22B	Reflection standard
7	S22C	Reflection standard
8	S12T	Thru/Delay standard

3-port analyzers only

9	S33A	Reflection standard
10	S33B	Reflection standard
11	S33C	Reflection standard
12	S32T	Thru/Delay standard
13	S23T	Thru/Delay standard
14	S31T	Thru/Delay standard
15	S13T	Thru/Delay standard

TRL Calibrations

16	TRL "T"	Thru standard
17	TRL "R"	Reflect standard
18	TRL "L"	Line standard

<name> Label for the calibration class. Must be enclosed in quotes. Any string between 1 and 12 characters long. Cannot begin with a numeric.

Examples

```
SENS:CORR:COLL:CKIT:OLAB3 'LOADS'
```

```
sense:correction:collect:ckit:olabel14 'Thru'
```

Return Type String

Default Not Applicable

SENSe:CORRection:COLLection:CKIT:OLIST[class]?

(Read-only) Returns seven values of standards that are assigned to the specified class.

This command ALWAYS applies to the Cal Kit that is selected (using [SENS:CORR:COLL:CKIT:SEL](#)) when this ORDER command is sent.

Parameters

<class> Number of the calibration class to be queried. The <class> numbers are associated with the following calibration Classes:

	Class	Description
1	S11A	Reflection standard
2	S11B	Reflection standard
3	S11C	Reflection standard
4	S21T	Thru/Delay standard
5	S22A	Reflection standard
6	S22B	Reflection standard
7	S22C	Reflection standard
8	S12T	Thru/Delay standard

3-port analyzers ONLY

4-port analyzers use S11 and S22 classes ([see example program](#))

9	S33A	Reflection standard
10	S33B	Reflection standard
11	S33C	Reflection standard
12	S32T	Thru/Delay standard
13	S23T	Thru/Delay standard
14	S31T	Thru/Delay standard
15	S13T	Thru/Delay standard

TRL Calibrations

16	TRL "T"	Thru standard
17	TRL "R"	Reflect standard
18	TRL "L"	Thru standard

Examples

```
SENS:CORR:COLL:CKIT:OLIST8?
```

Always returns 7 standard numbers. Unassigned standards return 0

Return Type

Numeric; returns the <class> number of the selected standard.

Default

Not Applicable

SENSe:CORRection:COLLect:CKIT:ORDeR<class> <std> [,<std>] [,<std>] [,<std>] [,<std>] [,<std>]

(Read-Write) Sets a standard number to a calibration class. This command does **NOT** set or dictate the order for measuring the standards. For more information, see Assigning Standards to a Calibration Class.

This command ALWAYS applies to the Cal Kit that is selected (using [SENS:CORR:COLL:CKIT:SEL](#)) when this ORDeR command is sent.

Parameters

<class> Number of the calibration class that is assigned to <standard>. Choose a number between: **1** and **18**. The <class> numbers are associated with the following calibration Classes:

	Class	Description
1	S11A	Reflection standard
2	S11B	Reflection standard
3	S11C	Reflection standard
4	S21T	Thru/Delay standard
5	S22A	Reflection standard
6	S22B	Reflection standard
7	S22C	Reflection standard
8	S12T	Thru/Delay standard

3-port analyzers ONLY

4-port analyzers use S11 and S22 classes ([see example program](#))

9	S33A	Reflection standard
10	S33B	Reflection standard
11	S33C	Reflection standard
12	S32T	Thru/Delay standard
13	S23T	Thru/Delay standard
14	S31T	Thru/Delay standard
15	S13T	Thru/Delay standard

TRL Calibration

16	TRL "T"	Thru standard
17	TRL "R"	Reflect standard
18	TRL "L"	Line standard

<std> Standard number to be assigned to the class; Choose a standard between 1 and 30. One standard is mandatory; up to six additional standards are optional.

Examples

```
'Assigns standard 3 to S11A class:  
SENS:CORR:COLL:CKIT:ORD1 3
```

```
'Assigns standard 2 and 5 to S21T class class:  
sense:correction:collect:ckit:order4 2,5
```

Query Syntax

SENSe:CORRection:COLLEct:CKIT:ORDeR<class>?

'Returns only the first standard assigned to the specified class. To query the remaining standards, use

[SENSe:CORRection:COLLEct:CKIT:OLIST\[1-15\]?](#)

Return Type

Numeric

Default

Not Applicable

SENSe<num>:CORRection:COLLEct:CKIT:PORT<n>[:SELEct] <string>

(Read-Write) Sets and returns the name of the Cal Kit to use for **Unguided** cal.

This command effectively does the same task as [SENS:CORR:COLL:CKIT](#) but specifies the cal kit by name.

Note: During an [Unguided cal](#), you can access ONLY mechanical cal kits #1 through #95. However, there is no limit to the number of cal kits that can be imported.

Parameters

<num> Currently not used. The unguided cal kit selection is for all ports on all channels.

<n> Currently not used. The unguided cal kit selection is for all ports on all channels.

<string> Cal Kit name enclosed in quotes. Use [SENS:CORR:COLL:CKIT:CAT?](#) to read a list of all available Cal Kits in the PNA.

Examples

```
SENS:CORR:COLL:CKIT:PORT "85052B"  
sense2:correction:collect:ckit:port:select "85052D"
```

Query Syntax

SENSe<num>:CORRection:COLLEct:CKIT:PORT<n>:SELEct?

Return Type

String

Default

Last kit selected

SENSe:CORRection:COLLEct:CKIT:RESet <num> - Superseded

This command is replaced by [Sens:Corr:Ckit:Init](#).

(Write-only) Resets the selected calibration kit to factory default definition values.

Parameters

<num> The number of the calibration kit to be reset. Choose any integer between: **1** and **8**

Examples

```
SENS:CORR:COLL:CKIT:RESet 1  
sense:correction:collect:ckit:reset 4
```

Query Syntax Not Applicable

Default Not Applicable

SENSe<cnum>:CORRection:COLLEct:CKIT[:SELEct] <num>

(Read-Write) Selects (makes active) a calibration kit for **performing** an **UNGUIDED** calibration or for **modifying** standards. All subsequent "CKIT" commands that are sent apply to this selected calibration kit. Select a calibration standard using [SENS:CORR:COLL:CKIT:STAN <num>](#). Kits 1 to approximately kit 37 are factory installed Cal Kits.

Note: During an [Unguided cal](#), you can access **ONLY** mechanical cal kits #1 through #95. However, there is no limit to the number of cal kits that can be imported.

This command effectively does the same task as [SENS:CORR:COLL:CKIT:PORT](#) which specifies the cal kit by name instead of this command which specifies by number.

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1.

<num> The number of the calibration kit. Choose from:

Use [SENSe:CORRection:COLLEct:CKIT:RESet](#) to restore Cal Kits to default values.

Name

- 1 Cal Kit 1
- 2 Cal Kit 2
- 3 Cal Kit 3
- "
- "
- 94 Cal Kit 94

- 95 Cal Kit 95
- 99 ECal module

Examples `SENS:CORR:COLL:CKIT 2`
`sense2:correction:collect:ckit:select 7`

Query Syntax `SENSe<num>:CORRection:COLLect:CKIT?`

Return Type Numeric

Default Last kit selected

SENSe:CORRection:COLLect:CKIT:STANdard:C0 <num>

(Read-Write) Sets the C0 value (the first capacitance value) for the selected standard. For a detailed discussion of this value, search for App Note 8510-5B at www.Agilent.com.

Parameters

<num> Value for C0 in femtofarads (1E-15)

Examples The following commands set C0=15 femtofarads:

```
SENS:CORR:COLL:CKIT:STAN:C0 15
sense:correction:collect:ckit:standard:c0 15
```

Query Syntax `SENSe:CORRection:COLLect:CKIT:STANdard:C0?`

Return Type Numeric

Default Not Applicable

SENSe:CORRection:COLLect:CKIT:STANdard:C1 <num>

(Read-Write) Sets the C1 value (the second capacitance value) for the selected standard. For a detailed discussion of this value, search for App Note 8510-5B at www.Agilent.com.

Parameters

<num> Value for C1.

Examples The following two commands set C1=15:

```
SENS:CORR:COLL:CKIT:STAN:C1 15
sense:correction:collect:ckit:standard:c1 15
```

Query Syntax `SENSe:CORRection:COLLect:CKIT:STANdard:C1?`

Return Type Numeric

Default Not Applicable

SENSe:CORRection:COLLEct:CKIT:STANdard:C2 <num>

(Read-Write) Sets the C2 value (the third capacitance value) for the selected standard. For a detailed discussion of this value, search for App Note 8510-5B at www.Agilent.com.

Parameters

<num> Value for C2.

Examples

The following two commands set C2:

```
SENS:CORR:COLL:CKIT:STAN:C2 15  
sense:correction:collect:ckit:standard:c2 15
```

Query Syntax SENSe:CORRection:COLLEct:CKIT:STANdard:C2?

Return Type Numeric

Default Not Applicable

SENSe:CORRection:COLLEct:CKIT:STANdard:C3 <num>

(Read-Write) Sets the C3 value (the fourth capacitance value) for the selected standard. For a detailed discussion of this value, search for App Note 8510-5B at www.Agilent.com.

Parameters

<num> Value for C3.

Examples

The following two commands set C3

```
SENS:CORR:COLL:CKIT:STAN:C3 15  
sense:correction:collect:ckit:standard:c3 15
```

Query Syntax SENSe:CORRection:COLLEct:CKIT:STANdard:C3?

Return Type Numeric

Default Not Applicable

SENSe:CORRection:COLLEct:CKIT:STANdard:CHARacter <char>

(Read-Write) Sets the media type of the selected calibration standard.

Parameters

<char> Media type of the standard. Choose from:
Coax - Coaxial Cable
Wave - Waveguide

Examples

```
SENS:CORR:COLL:CKIT:STAN:CHAR COAX  
sense:correction:collect:ckit:standard:character wave
```

Query Syntax SENSE:CORRection:COLLect:CKIT:STANdard:CHARacter?

Return Type Numeric

Default Coax

SENSe:CORRection:COLLect:CKIT:STANdard:DELay <num>

(Read-Write) Sets the electrical delay value for the selected standard.

Parameters

<num> Electrical delay in picoseconds

Examples

```
The following two commands set delay to 50 picoseconds  
SENS:CORR:COLL:CKIT:STAN:DEL 50e-12  
sense2:correction:collect:ckit:standard:delay 50ps
```

Query Syntax SENSE:CORRection:COLLect:CKIT:STANdard:DELay?

Return Type Numeric

Default Not Applicable

SENSe:CORRection:COLLect:CKIT:STANdard:FMAXimum <num>

(Read-Write) Sets the maximum frequency for the selected standard.

Parameters

<num> Maximum frequency in Hertz.

Examples

```
SENS:CORR:COLL:CKIT:STAN:FMAX 9e9  
sense:correction:collect:ckit:standard:fmaximum 9Ghz
```

Query Syntax SENSE:CORRection:COLLEct:CKIT:STANdard:FMAXimum?

Return Type Numeric

Default Not Applicable

SENSe:CORRection:COLLEct:CKIT:STANdard:FMINimum <num>

(Read-Write) Sets the minimum frequency for the selected standard.

Parameters

<num> Minimum frequency in Hertz.

Examples

```
SENS:CORR:COLL:CKIT:STAN:FMIN 1e3  
sense:correction:collect:ckit:standard:fminimum 1khz
```

Query Syntax SENSE:CORRection:COLLEct:CKIT:STANdard:FMINimum?

Return Type Numeric

Default Not Applicable

SENSe:CORRection:COLLEct:CKIT:STANdard:IMPedance <num>

(Read-Write) Sets the characteristic impedance for the selected standard.

Parameters

<num> Impedance in Ohms

Examples

```
SENS:CORR:COLL:CKIT:STAN:IMP 75  
sense:correction:collect:ckit:standard:impedance 50.3
```

Query Syntax SENSE:CORRection:COLLEct:CKIT:STANdard:IMPedance?

Return Type Numeric

Default 50

SENSe:CORRection:COLLEct:CKIT:STANdard:L0 <num>

(Read-Write) Sets the L0 value (the first inductance value) for the selected standard. For a detailed discussion of this value, search for App Note 8510-5B at www.Agilent.com.

Parameters

<num> Value for L0 in femtohenries (1E-15)

Examples

The following two commands set L0=15 femtohenries:

```
SENS:CORR:COLL:CKIT:STAN:L0 15
sense:correction:collect:ckit:standard:l0 15
```

Query Syntax

SENSe:CORRection:COLLect:CKIT:STANdard:L0?

Return Type

Numeric

Default

Not Applicable

SENSe:CORRection:COLLect:CKIT:STANdard:L1 <num>

(Read-Write) Sets the L1 value (the second inductance value) for the selected standard. For a detailed discussion of this value, search for App Note 8510-5B at www.Agilent.com.

Parameters

<num> Value for L1.

Examples

The following two commands set L1=15:

```
SENS:CORR:COLL:CKIT:STAN:L1 15
sense:correction:collect:ckit:standard:l1 15
```

Query Syntax

SENSe:CORRection:COLLect:CKIT:STANdard:L1?

Return Type

Numeric

Default

Not Applicable

SENSe:CORRection:COLLect:CKIT:STANdard:L2 <num>

(Read-Write) Sets the L2 value (the third inductance value) for the selected standard. For a detailed discussion of this value, search for App Note 8510-5B at www.Agilent.com.

Parameters

<num> Value for L2.

Examples

The following two commands set L2=15:

```
SENS:CORR:COLL:CKIT:STAN:L2 15
sense:correction:collect:ckit:standard:l2 15
```

Query Syntax

SENSe:CORRection:COLLect:CKIT:STANdard:L2?

Return Type

Numeric

Default

Not Applicable

SENSe:CORRection:COLLect:CKIT:STANdard:L3 <num>

(Read-Write) Sets the L3 value (the fourth inductance value) for the selected standard. For a detailed discussion of this value, search for App Note 8510-5B at www.Agilent.com.

Parameters

<num> Value for L3.

Examples

The following two commands set L3=15:

```
SENS:CORR:COLL:CKIT:STAN:L3 15
sense:correction:collect:ckit:standard:l3 15
```

Query Syntax

SENSe:CORRection:COLLect:CKIT:STANdard:L3?

Return Type

Numeric

Default

Not Applicable

SENSe:CORRection:COLLect:CKIT:STANdard:LABel <name>

(Read-Write) Sets the label for the selected standard. The label is used to prompt the user to connect the specified standard.

Parameters

<name> Label for the standard; Must be enclosed in quotes. Any string between **1** and **12** characters long. Cannot begin with a numeric.

Examples

```
SENS:CORR:COLL:CKIT:STAN:LAB 'OPEN'  
sense:correction:collect:ckit:standard:label 'Short2'
```

Query Syntax SENSE:CORRection:COLLect:CKIT:STANdard:LABel?

Return Type String

Default Not Applicable

SENSe:CORRection:COLLect:CKIT:STANdard:LOSS <num>

(Read-Write) Sets the insertion loss for the selected standard.

Parameters

<num> Insertion loss in Gohms / sec. (GigaOhms per second of electrical delay)

Examples

```
SENS:CORR:COLL:CKIT:STAN:LOSS 3.5e9  
sense:correction:collect:ckit:standard:loss 3
```

Query Syntax SENSe:CORRection:COLLect:CKIT:STANdard:LOSS?

Return Type Numeric

Default Not Applicable

SENSe:CORRection:COLLect:CKIT:STANdard:REMOve

(Write only) Deletes the selected standard from the selected cal kit.

Examples

```
SENS:CORR:COLL:CKIT:STAN:REMOve
```

Default Not Applicable

SENSe:CORRection:COLLect:CKIT:STANdard:SDEscription <string>

(Read-Write) Modifies the description of the selected standard of the selected kit. This description appears in the [edit kit dialog box](#).

Parameters

<string> Description of the standard.

Examples `SENS:CORR:COLL:CKIT:STAN:SDES "My New Standard"`

Query Syntax `SENSe:CORRection:COLLect:CKIT:STANdard:SDEscription?`

Return Type String

Default Not Applicable

SENSe:CORRection:COLLect:CKIT:STANdard[:SELEct] <num>

(Read-Write) Selects the calibration standard. All subsequent "CKIT" commands to modify a standard will apply to the selected standard. Select a calibration kit using [SENS:CORR:COLL:CKIT:SEL](#)

Parameters

<num> Number of the standard. Choose any number between:
1 and 30

Examples `SENS:CORR:COLL:CKIT:STAN 3`
`sense:correction:collect:ckit:standard:select 8`

Query Syntax `SENSe:CORRection:COLLect:CKIT:STANdard[:SELEct]?`

Return Type Numeric

Default 1

SENSe:CORRection:COLLect:CKIT:STANdard:TYPE <char>

(Read-Write) Sets the type for the selected standard.

Parameters

<char> Choose from:
OPEN
SHORT
LOAD
SLOAD (sliding load)

THRU (through)

ARBI(arbitrary)

Examples

```
SENS:CORR:COLL:CKIT:STAN:TYPE LOAD  
sense:correction:collect:ckit:standard:type short
```

Query Syntax

SENSe:CORRection:COLLEct:CKIT:STANdard:TYPE?

Return Type

Character

Default

Not Applicable

SENSe:CORRection:COLLEct:CKIT:STANdard:TZReal <num>

(Read-Write) Sets the TZReal component value of the Terminal Impedance for the selected standard.

Note: Only applicable when the Standard Type is set to **ARBI**

Parameters

<num> Value for TZReal in Ohms

Examples

The following commands set TZReal=15 Ohms:

```
SENS:CORR:COLL:CKIT:STAN:TZReal 15  
sense:correction:collect:ckit:standard:TZReal 15
```

Query Syntax

SENSe:CORRection:COLLEct:CKIT:STANdard:TZReal?

Return Type

Numeric

Default

Not Applicable

SENSe:CORRection:COLLEct:CKIT:STANdard:TZImag <num>

(Read-Write) Sets the TZImag component value of the Terminal Impedance for the selected standard.

Note: Only applicable when the Standard Type is set to **ARBI**

Parameters

<num> Value for TZImag in Ohms

Examples

The following two commands set TZImag=15 Ohms:

```
SENS:CORR:COLL:CKIT:STAN:TZImag 15
sense:correction:collect:ckit:standard:TZImag 15
```

Query Syntax

SENSe:CORRection:COLLect:CKIT:STANdard:TZImag?

Return Type

Numeric

Default

Not Applicable

SENSe:CORRection:COLLect:CKIT:TRLOption:IMPedance <char>

(Read-Write) Sets the reference impedance when using this TRL cal kit. [Learn more](#).

Before sending this command, select a cal kit using [SENS:CORR:COLL:CKIT:SElect](#).

Parameters

<char> Choose from:

SYSTEM - The system impedance is used as the reference impedance. During a Guided or Unguided Cal, the Z0 of the Cal standard's connector definition sets the System Z0.

Make this selection when the desired test port impedance differs from the impedance of the LINE standard. Also, make this selection when skin effect impedance correction is desired for coax lines.

LINE The impedance of the line standard is used as the reference impedance, or center of the Smith Chart. Any reflection from the line standard is assumed to be part of the directivity error.

Examples

```
SENS:CORR:COLL:CKIT:TRL:IMP SYST
sense:correction:collect:ckit:trloption:impedance line
```

Query Syntax

SENSe:CORRection:COLLect:CKIT:TRLOption:IMPedance?

Return Type

Character

Default

LINE

SENSe:CORRection:COLLect:CKIT:TRLOption:LRLChar <bool>

(Read-Write) This setting ONLY applies if an LRL Cal Kit is being modified AND Testport Reference Plane is set to THRU AND the TRL Thru class standard and the TRL Line/Match class standard both have the same values for Offset Z0 and Loss. Otherwise, this setting is ignored.

Before sending this command, select a cal kit using [SENS:CORR:COLL:CKIT:SElect](#).

Parameters

<bool> Choose from:

1 or **ON** - Automatically correct for line loss and dispersion characteristics.

0 or **OFF** - Select when anomalies appear during a calibrated measurement which may indicate different loss and impedance values for the Line standards.

Examples

```
SENS:CORR:COLL:CKIT:TRL:LRLC 1
```

```
sense:correction:collect:ckit:trloption:lrlchar off
```

Query Syntax SENSe:CORRection:COLLect:CKIT:TRLOption:LRLChar?

Return Type Boolean

Default OFF

SENSe:CORRection:COLLect:CKIT:TRLOption:RPLane <char>

(Read-Write) Sets the reference impedance when using this cal kit. [Learn more](#).

Before sending this command, select a cal kit using [SENS:CORR:COLL:CKIT:SElect](#).

Parameters

<char> Choose from:

THRU The THRU standard definition is used to establish the measurement reference plane. Select if the THRU standard is zero-length or very short.

REFlect The REFLECT standard definition is used to establish the position of the measurement reference plane. Select if the THRU standard is not appropriate AND the delay of the REFLECT standard is well defined. Also, select If a flush short is used for the REFLECT standard because a flush short provides a more accurate phase reference than a Thru standard.

Examples

```
SENS:CORR:COLL:CKIT:TRL:RPL THRU
```

```
sense:correction:collect:ckit:trloption:rplane reflect
```

Query Syntax	SENSe:CORRection:COLLect:CKIT:TRLOption:RPLane?
Return Type	Character
Default	THRU

Last Modified:

5-Nov-2013	Edit to ORD command. Extend to 30 stds.
16-Aug-2013	Updated Ecal nums
21-Dec-2012	Added TRL commands
12-Oct-2011	Select Cal kit edits
11-Jan-2011	Minor edits
17-Mar-2010	Added Ckit:CAT and CKIT:Select by string (9.2)
10-Nov-2008	Clarified Select command
30-Oct-2008	Fixed SDES query per CN
14-Apr-2008	Added link to app note
17-Sep-2007	Fixed 'select' command

Sense:Correction:Cset Commands

Performs actions on calibration sets.

SENSe:CORRection:CSET

ACTivate

CATalog?

COPY

CREate

DATA

DELete

DESCription

ETERm

| **CATalog?**

| **[DATA]**

FLATten

GUID

NAME

SAVE

[SElect]

STANdard

STIMulus?

TSET

| **ALLPorts?**

| **TYPE?**

TYPE

Click on a [blue](#) keyword to view the command details.

Red keywords are superseded commands. [Learn more](#).

See Also

- [Creating Cal Sets](#)
- [Example Programs](#)
- [Learn about Cal Sets](#)
- [Synchronizing the PNA and Controller](#)

SENSe<cnum>:CORRection:CSET:ACTivate <string>, <bool>

This command replaces [SENS:CORR:CSET:GUID](#)

(Read-Write) Selects and applies a Cal Set to the specified channel.

Use [SENS:CORR:CSET:CAT?](#) to list the Cal Sets.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <string> Cal Set to make active. Specify the Cal Set by **GUID** or **Name**. Use [SENS:CORR:CSET:CAT?](#) to list the available Cal Sets in either format.
- <bool> Should the Cal Set stimulus values be applied to the channel. Choose from:
 - ON (1)** Apply the Cal Set stimulus values to the channel.
 - OFF (0)** Do NOT apply the Cal Set stimulus values. If the Cal Set stimulus values do not match the channel stimulus values, then the following will occur:
 - If interpolation is ON, then interpolation will be attempted. This may fail if the channel frequency is outside the range of the Cal Set.
 - If interpolation is OFF, the selection will be abandoned and an error is returned:

Examples

```
SENS:CORR:CSET:ACT "My2Port",1  
  
sense:correction:cset:activate? name  
'returns  
"My2Port"
```

Query Syntax SENSe<cnum>:CORRection:CSET:ACTivate? [GUID|NAME]

Returns the name of the Cal Set that is applied to the specified channel. Choose from **GUID** or **NAME** to specify which string is returned. If unspecified, the GUID of the Cal Set is returned. If no Cal Set is applied to the specified channel, then "No Calset Selected" is returned.

Return Type String

Default Not Applicable

SENSe:CORRection:CSET:CATalog? [char] - Superseded

This command is replaced by [CSET:CAT?](#)

(Read-only) Returns a list of Cal Sets.

Parameters

<char> Optional argument. The list is returned in one of the following formats. Both return comma-separated string lists.

GUID Cal Sets are listed by GUID (Default if unspecified).

NAME Cal Sets are listed by Name

Examples

```
SENS:CORR:CSET:CAT?
```

```
'Returns:
```

```
{FD6F863E-9719-11d5-8D6C-00108334AE96},{1B03B2CE-971A-11d5-8D6C-00108334AE96}
```

```
sense2:correction:cset:catalog? name
```

Default Not Applicable

SENSe<num>:CORRection:CSET:COPY <string>

(Write-only) Creates a new Cal Set and copies the current Cal Set data into it. Use this command to manipulate data on a Cal Set without corrupting the original cal data.

Parameters

- <cnum> Channel number using the Cal Set to be copied. If unspecified, value is set to 1
- <string> Name of the new Cal Set.

Examples `SENS2:CORR:CSET:COPY 'My2Port'`

Query Syntax Not Applicable

Default Not Applicable

SENSe<cnum>:CORRection:CSET:CREate [name]

(Write-only) Creates an empty Cal Set and attaches it to the specified channel. This command is ONLY necessary before remotely filling the Cal Set with error term data. (For Advanced Users).

A Cal Set is automatically created, applied to the channel, and saved at the completion of a guided cal according to the preference setting [SENS:CORR:PREF:CSET:SAVE](#).

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- [name] Optional argument. Name of the Cal Set. Spaces or punctuation are NOT allowed. If unspecified, a unique name is chosen in the form "Calset_N" where N is a unique number.

Examples `SENS:CORR:CSET:CRE 'My2Port'`

Query Syntax Not Applicable

Default Not Applicable

SENSe<cnum>:CORRection:CSET:DATA <eterm, portA, portB,>[<rec>] <block>

(Read-Write) Read or Write a specific error term from/to the Cal Set currently attached to the specified channel. (For Advanced Users). The command can be used only for the error terms listed. See [SENS:CORR:CSET:ETERM](#) to get and put error term data using a string argument for all error terms.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <eterm, portA, portB,> **Error Term, Port pair of the specified error term.**
- portR>

portB

Although not all error terms use two port numbers, two are required by the PNA in all cases. Each port number must be between 1 and the number of ports on the PNA.

EDIR - directivity

portA: the port at which directivity is measured.

portB: Not used, but must be a valid PNA port number.

ESRM - source match

portA: the port at which source match is measured.

portB: Not used, but must be a valid PNA port number.

ERFT - reflection tracking

portA: the port at which reflection tracking is measured.

portB: Not used, but must be a valid PNA port number.

ELDM - load match

portA: the port at which load match is measured.

portB: the source port.

Load match is measured with a cable connected between the measured port (portA) and the source port (portB).

The cal system requires that the complete matrix of loadmatch arrays be filled.

In most cases you can measure loadmatch once at a port, driven by any other port. Then use that data for all variations of the receive port. (The exception is the 3-port PNA models, which requires the loadmatch-measured port to be driven by every other port.)

For example: Measure the loadmatch at port2 while driving port1. Then upload this same data to the following arrays:

ELDM,2,1,<data>

ELDM,2,3,<data>

ELDM,2,4,<data>

ETRT - transmission tracking

portA: the receive port

portB: the source port for this measurement

EXTLK - crosstalk

portA: the receive port

portB: the source port for this measurement

ERSPT - response tracking.

portA: Not used, but must be a valid PNA port number.

portB: Not used, but must be a valid PNA port number.

ERSPI - response isolation.

portA: Not used, but must be a valid PNA port number.

portB: Not used, but must be a valid PNA port number.

<rec> <**string**> - Specify the PNA receiver for which the Eterm applies.

Required ONLY when Eterm is response tracking (**ERSPT**) or response isolation (**ERSPI**).

[Logical receiver notation](#) is allowed.

A full 4-port calibration requires the following terms be uploaded:

		PORT B			
		1	2	3	4
P O R	1	EDIR,1,1	ELDM,1,2	ELDM,1,3	ELDM,1,4
		ERFT,1,1	ETRT,1,2	ETRT,1,3	ETRT,1,4
		ESRM,1,1	EXTLK,1,2	EXTLK,1,3	EXTLK,1,4
	2	ELDM,2,1	EDIR,2,2	ELDM,2,3	ELDM,2,4
		ETRT,2,1	ERFT,2,2	ETRT,2,3	ETRT,2,4
		EXTLK,2,1	ESRM,2,2	EXTLK,2,3	EXTLK,2,4

T	3	ELDM,3,1	ELDM,3,2	EDIR,3,3	ELDM,3,4
		ETRT,3,1	ETRT,3,2	ERFT,3,3	ETRT,3,4
EXTLK,3,1		EXTLK,3,2	ESRM,3,3	EXTLK,3,4	
A	4	ELDM,4,1	ELDM,4,2	ELDM,4,3	EDIR,4,4
		ETRT,4,1	ETRT,4,2	ETRT,4,3	ERFT,4,4
		EXTLK,4,1	EXTLK,4,2	EXTLK,4,3	ESRM,4,4

Reflection terms

Transmission terms

<block> (Block). Error term data. A Real / Imaginary data pair for each data point.

Format is set using [FORM:DATA](#) command.

For REAL binary formats, refer to [Getting Data from the Analyzer using SCPI](#)

Example

```
'Set the directivity term with a cal set using 5 points
SENS1:CORR:CSET:DATA EDIR, 1, 1, +6.125696000000E-002,-
7.271638000000E-003,-3.638120000000E-003,+1.335218000000E-002,-
4.367751000000E-003,+1.877924000000E-002,-4.092391000000E-
003,+4.242912000000E-002,-2.037849000000E-002,+3.214251000000E-002"
```

Query Syntax

SENSe<cnum>:CORRection:CSET:DATA? <eterm,portA, portB>,<rec>

Query Examples

```
'Read the response isolation eterms for the port 1 reference
receiver
sens:corr:cset:data? ERSPI,1,1,'R1'
'Same receiver using logical receiver notation
sens:corr:cset:data? ERSPI,1,1,'a1'
```

Return Type

Block data

Default

Not Applicable

SENSe:CORRection:CSET:DELeTe <string> - Superseded

This command is replaced by [CSET:DEL](#).

(Write-only) Deletes a Cal Set from the set of available Cal Sets. This method immediately updates the Cal Set file on the hard drive. If the Cal Set is currently being used by a channel or does not exist, this request will be denied and an error is returned.

Parameters

<string> Cal Set to be deleted. Specify the Cal Set by **GUID** or **Name**. Use [SENS:CORR:CSET:CAT?](#) to list the available Cal Sets in either format.

Examples

```
SENS:CORR:CSET:DEL '{2B893E7A-971A-11d5-8D6C-00108334AE96}'  
sense2:correction:cset:delete 'MyCalSet'
```

Query Syntax Not Applicable

Default Not Applicable

SENSe<num>:CORRection:CSET:DESCRiption <string>

(Read-Write) Sets or returns the descriptive string assigned to the selected Cal Set. Change this string so that you can easily identify each Cal Set. Apply and select the Cal Set using [SENS:CORR:CSET:ACT](#).

Parameters

<num> Any existing channel number. If unspecified, value is set to 1

<string> The descriptive string associated with the currently-selected Cal Set

Examples

```
SENS:CORR:CSET:DESC 'MyCalSet'  
sense2:correction:cset:description 'thisCalSet'
```

Query Syntax SENSe<num>:CORRection:CSET:DESCRiption?

Return Type String

Default Not Applicable

SENSe<num>:CORRection:CSET:ETERm[:DATA] <string>,<r, i [r,i]...>

(Read-Write) Sets or returns error term data for all PNA measurements.

Parameters

- <num> Any existing channel number. If unspecified, value is set to 1
- <string> (String) Error term to read or write. The error term is specified using the EXACT case-sensitive string displayed in the [Cal Set Viewer](#) utility. See [SENS:CORR:CSET:DATA](#) for a description of port numbers.

The following Noise figure error terms are listed for convenience:

- **RcvNoiseCorr_m_n** Noise correlation matrix of the noise receiver (a 2x2 complex matrix). The row and column indices m and n range from 1 to 2.
- **RcvT_m_n** T-matrix of the noise receiver (a 2x2 complex matrix). The row and column indices m and n range from 1 to 2.
- **GammaTuner_n** Reflection coefficient for impedance state n of the embedded noise tuner (Ecal module) in the port 1 source path. For the Agilent 4691 family of Ecal modules, n can range from 1 to 7.

<r,i> (Block) Error term data. A Real / Imaginary data pair for each data point.

Format is set using [FORM:DATA](#) command.

For REAL binary formats, refer to [Getting Data from the Analyzer using SCPI](#)

Examples

```
SENS:CORR:CSET:ETERM "Directivity(1,1)", 0.237,-1.422, 0.513,  
0.895 ' set directivity(source error term for 2 points  
SENS:CORR:CSET:ETERM? "Directivity(1,1)" 'read
```

Query Syntax SENSE<num>:CORRection:CSET:ETERm[:DATA]? <string>

Return Type Block data

Default Not Applicable

SENSe<num>:CORRection:CSET:ETERm:CATalog?

(Read-only) Returns a list of error term names found in the current Cal Set that is applied to the specified channel.

Parameters

Examples

```
SENS:CORR:CSET:ETER:CAT?  
  
'For a 1-port cal, returns  
  
"Directivity(1,1),ReflectionTracking(1,1),SourceMatch(1,1)"
```

Return Type String

Default Not Applicable

SENSe<num>:CORRection:CSET:FLATten <string>

(Write-only) When a Cal Set that was produced by a calibration has been interpolated or otherwise modified (for example, by [Fixturing operations](#)) this command saves the modified Cal Set to the PNA hard drive so that it can be reused. There is no User Interface equivalent for this command.

Background

When a Cal Set is selected for use by a channel, the channel reads the Cal Set from disk (master Cal Set). If the channel aligns perfectly with the Cal Set, the master Cal Set is used directly. In this case, the active Cal Set is the master Cal Set.

When processing occurs on the error terms due to interpolation or modification due to the use of fixturing, the channel will generate a temporary "memory-resident" Cal Set. In this case, the active Cal Set is the memory-resident Cal Set. This FLATten command allows you to save the active Cal Set to disk.

Depending on the measurement conditions, this flattening of the Cal Set can improve performance, especially if the Cal Set is applied often (using multiple recall states) or used by many channels. Flattening a version of the Cal Set for each channel can avoid the interpolation or the fixturing processing that would otherwise occur when the Cal Set is selected or the instrument state is recalled.

You will have to manage the application of such a Cal Set as the PNA itself will have no way to determine what processing had been done once the flatten command is used. For example, if fixture de-embedding occurred prior to the flatten command, that Cal Set should then be applied WITHOUT fixturing on, because fixturing is already embedded in that Cal Set. It is your responsibility to apply the Cal Set properly.

If you want to repeatedly de-embed multiple networks (i.e. concatenate multiple 2-port de-embedding files) you can use the flatten command to create a new master Cal Set after each de-embed, and

sequentially add additional de-embed networks.

Parameters

- <cnun> Channel number on which the modified Cal Set resides. If unspecified, value is set to 1
- <string> Name of the new Cal Set. Spaces or punctuation NOT allowed.

Examples

```
SENS:CORR:CSET:FLAT "MyCalSet"
```

Query Syntax Not Applicable

Default Not Applicable

SENSe<cnun>:CORRection:CSET:GUID <string> Superseded

This command is replaced by [SENS:CORR:CSET:ACTivate](#).

(Read-Write) Selects the Cal Set identified by the string parameter (GUID) and applies it to the specified channel.

- A Cal Set cannot be selected for a channel which is not ON.
- If the stimulus settings of the selected Cal Set differ from those of the selected channel, the instrument will automatically change the channel's settings to match the Cal Set.

Parameters

- <cnun> Any existing channel number. If unspecified, value is set to 1
- <string> GUID of the desired Cal Set. The curly brackets and hyphens must be included.

Examples

```
SENS:CORR:CSET:GUID '{2B893E7A-971A-11d5-8D6C-00108334AE96}'  
sense2:correction:cset:guid '{2B893E7A-971A-11d5-8D6C-  
00108334AE96}'
```

Query Syntax SENSe<cnun>:CORRection:CSET:GUID?

Returns the GUID of the currently-selected Cal Set for the specified channel.

Return Type String

Default Not Applicable

SENSe<cnun>:CORRection:CSET:NAME <string>

(Read-Write) Sets or queries the name of the Cal Set currently applied to the specified channel.

Parameters

- <cnun> Any existing channel number. If unspecified, value is set to 1
- <string> Name of the Cal Set. Spaces or punctuation NOT allowed.

Examples

```
SENS:CORR:CSET:NAME 'MyCalSet'  
sense2:correction:cset:name 'thisCalSet'
```

Query Syntax SENSE<cnun>:CORRection:CSET:NAME?

Return Type String

Default Not Applicable

SENSe<cnun>:CORRection:CSET:SAVE [<char>]

This command is NOT necessary after completion of a calibration. A Cal Set is automatically created, applied to the channel, and saved at the completion of a guided cal according to the preference setting [SENS:CORR:PREF:CSET:SAVE](#).

(Read Write)

Saves the channel's Cal Set to the PNA hard drive. For example, use this command after writing data to a Cal Set using [SENS:CORR:CSET:DATA](#) (For Advanced Users).

The file name is saved as "CSETx.cst" where x is the user number assigned to <char>, and .cst specifies a Cal Set and instrument state. This is not the same syntax as a file saved through the default choices from the front panel, which is "at00x.cst". For more information on the file naming syntax, see the [MMEMory](#) subsystem. Learn more about [Instrument/Cal States](#).

Parameters

- <cnun> Any existing channel number. If unspecified, value is set to 1
- [<char>] Optional argument. Choose from:

USER01

USER02...

and so forth, until...

USER10

If <char> is NOT specified, changes that may have been made are saved to the cal set and NOT to the *.cst file.

Examples

```
SENS:CORR:CSET:SAVE USER03
sense2:correction:cset:save user09

'save changes to only the cal set

SENS:CORR:CSET:SAVE
```

Query Syntax SENSE<cnum>:CORRection:CSET:SAVE?

Queries the last correction set saved.

Return Type Character**Default** Not applicable**SENSe<cnum>:CORRection:CSET[:SElect] <char> Superseded**This command is replaced by [MMEM:LOAD](#)

(Read-Write) Recalls a *.cst file from memory. The file name is "CSETx.cst" where x is the user number assigned to <char>. Learn more about [.cst files](#)

For more information on the file naming syntax, see the [MMEMory](#) subsystem.

Note: This command does NOT select a Cal Set for a channel. To select a Cal Set, use [SENS:CORR:CSET:ACTivate](#)

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<char> Choose from:

DEF - Presets the analyzer**USER01** - Restores User01 calibration data**USER02** - Restores User02 calibration data

through...

USER10 - Restores User10 calibration data**Examples**

```
SENS:CORR:CSET DEF
sense2:correction:cset:select user02
```

Query Syntax SENSE<cnum>:CORRection:CSET[:SElect]?**Return Type** Character**Default** DEF

SENSe<num>:CORRection:CSET:STANdard <string>,<data>

(Read-Write) Sets or returns standard data. Standard data is available for Unguided Cals ONLY.

Parameters

- <num> Any existing channel number. If unspecified, value is set to 1
- <string> (String) Cal standard to read or write. The standard is specified using the EXACT case-sensitive string displayed in the [Cal Set Viewer](#) utility. See [SENS:CORR:CSET:DATA](#) for a description of port numbers.
- <data> (Block). Acquisition data. A Real / Imaginary data pair for each data point.

Format is set using [FORM:DATA](#) command.

For REAL binary formats, refer to [Getting Data from the Analyzer using SCPI](#)

Examples

```
SENS:CORR:CSET:STAN 'S11C(1,1), 0.237,-1.422, 0.513, 0.895 '
```

Set acquisition data for two points.

```
SENS:CORR:CSET:STAN? "S11C(1,1)" 'Read data
```

Query Syntax SENSe<num>:CORRection:CSET:STANdard? (string)

Return Type Block data

Default Not Applicable

SENSe<ch>:CORRection:CSET:STIMulus? [num]

(Read-only) Returns the source or response stimulus values for the Cal Set that is currently used by channel <ch>. Values are returned in the format specified by [FORM:DATA](#) (Block or ASCII).

Parameters

- <ch> Channel number to query Cal Set stimulus values. If unspecified, value is set to 1
- [num] Optional argument. Range of frequencies to return. These values would be different when FOM (Opt 080) is enabled.
 - 0 - returns source frequencies. Default setting if not specified.
 - 1 - returns response frequencies.
 - 2 - returns primary frequencies.

Examples

```
SENS:CORR:CSET:STIM?
```

```
sense:correction:cset:stimulus 1
```

Return Type Numeric

Default Not Applicable

SENSe:CORRection:CSET:TSET:ALLPorts? <cset>

(Read-only) Reads the port mapping used for the specified Cal Set. The returned values are the physical ports. The POSITION of the returned values corresponds to the logical ports.

For example, with an N44xx test set, if the returned string is "PNA 1,TS 2,PNA 2, TS 4" this means:

- PNA 1 is assigned to logical port 1
- TS 2 is assigned to logical port 2
- PNA 2 is assigned to logical port 3
- TS 4 is assigned to logical port 4

Parameters

<cset> **(String)** Name or GUID of the Cal Set. Use [SENS:CORR:CSET:CAT?](#) to read the list of available Cal Set names or GUIDs.

Examples

```
SENS:CORR:CSET:TSET:ALLP? "MyCalSet"  
sens:correction:cset:tset:allports? "{2B893E7A-971A-11d5-8D6C-  
00108334AE96}"
```

Return Type String

Default Not Applicable

SENSe:CORRection:CSET:TSET:TYPE? <cset>

(Read-only) Reads the test set type (model) used for the specified Cal Set.

Parameters

<cset> **(String)** Name or GUID of the Cal Set. Use [SENS:CORR:CSET:CAT?](#) to read the list of available Cal Set names or GUIDs.

Examples

```
SENS:CORR:CSET:TSET:TYPE? "MyCalSet"  
  
'returns "N44xx"  
  
sens:correction:cset:tset:type? "{2B893E7A-971A-11d5-8D6C-00108334AE96}"
```

Return Type String

Default Not Applicable

SENSe<ch>:CORRection:CSET:TYPE:CATalog? [format]

(Read-only) Query the Cal Types available in the selected Cal Set. The output is a comma separated list of Guids or a Cal Type names. [Learn more about applying Cal Types using SCPI.](#)

Use [CALC:CORR:TYPE](#) to apply a Cal Type.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1
[format] (Optional) Format of the output of cal types. choose from:
NAME - (default) returns a list of cal type string names.
GUID - returns a list of cal type GUIDs

Examples

```
SENS:CORR:CSET:TYPE:CAT? NAME  
SENS2:CORRection:CSET:TYPE:CAT?
```

Return Type String

Default Not Applicable

Last modified:

Added opt node to ETER
Modified CSET:ETER:CAT?
Modified Data command query
Optional <char> for save command
Added last paragraph to Flatten
Added CSET:ETerm:CAT?
Added Flatten command
Added <block> desc to cset:Stan command
Added <block> desc to cset:data command
Added Stimulus? (8.33)
Added CSET Delete by Name (8.0)
Added Noise ETerm and Std commands (8.0)

9/12/06 Added TSET commands for multiport.

Sense:Correction:Extension Commands

Performs and applies Port Extensions.

SENSe:CORRection:EXTension:

AUTO

- | **CONFig**
- | **DCOffset**
- | **LOSS**
- | **MEAS**
- | **PORT**
- | **RESet**
- | **STARt**
- | **STOP**

PORT

- | **DISTance**
- | **FREQ**
- | **INCLude**
 - | **[STATe]**
- | **LDC**
- | **LOSS**
- | **MEDium**
- | **SYSMedia**
- | **SYSVelocity**
- | **[TIME]**
- | **UNIT**

| [VELFactor](#)

| [WGCutoff](#)

RECEIVER

| [\[TIME\]](#)

[\[STATE\]](#)

Click on a [blue](#) keyword to view the command details.

See Also

- [Example Programs](#)
- [Learn about Port Extensions](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

SENSe<cnum>:CORRection:EXTension:AUTO:CONFIg <char>

(Read-Write) Sets the frequencies used to calculate Automatic Port Extension. [Learn more about calculating Automatic Port Extension.](#)

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<char> Frequencies to be used:

CSPN Use current frequency span.

AMKR - Use active marker frequency.

USPN - Use custom user span. Use [SENS:CORR:EXT:AUTO:STAR](#) and [SENS:CORR:EXT:AUTO:STOP](#) to specify start and stop frequency.

Examples

```
SENS:CORR:EXT:AUTO:CONF CSPN  
sense2:correction:extension:auto:config amkr
```

Query Syntax SENSe<cnum>:CORRection:EXTension:AUTO:CONFIg ?

Return Type Character

Default CSPN

SENSe<cnum>:CORRection:EXTension:AUTO:DCOFFset <bool>

(Read-Write) Specifies whether or not to include DC Offset as part of automatic port extension. [Learn more about Automatic DC Offset](#). Only allowed when [SENS:CORR:EXT:AUTO:LOSS](#) is set to ON.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<bool> ON (or 1) - Includes DC Offset correction.

OFF (or 0) - Does NOT include DC Offset correction.

Examples

```
SENS:CORR:EXT:AUTO:DCOF 1  
sense2:correction:extension:auto:dcoffset off
```

Query Syntax SENSe<cnum>:CORRection:EXTension:AUTO:DCOFFset?

Return Type Boolean

Default OFF (0)

SENSe<cnum>:CORRection:EXTension:AUTO:LOSS <bool>

(Read-Write) Specifies whether or not to include loss correction as part of automatic port extension. [Learn more about Loss Compensation](#) in port extension.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<bool> ON (or 1) - Includes Loss correction.

OFF (or 0) - Does NOT include Loss correction.

Examples

```
SENS:CORR:EXT:AUTO:LOSS 1  
sense2:correction:extension:auto:loss off
```

Query Syntax SENSe<cnum>:CORRection:EXTension:AUTO:LOSS?

Return Type Boolean

Default OFF (0)

SENSe<cnum>:CORRection:EXTension:AUTO:MEASure <char>

(Write-only) Measures either an OPEN or SHORT standard. When this command is sent, the PNA acquires the measurement with which to set automatic port extensions. [Learn more about which standard to measure.](#)

Parameters

<cnun> Any existing channel number. If unspecified, value is set to 1

<char> Standard to be measured. Choose from:

OPEN Measure OPEN standard

SHORT Measure SHORT standard

Examples

```
SENS:CORR:EXT:AUTO:MEAS OPEN  
sense2:correction:extension:auto:measure short
```

Query Syntax Not Applicable

Default Not Applicable

SENSe<cnun>:CORRection:EXTension:AUTO:PORT<n> <bool>

(Read-Write) Enables and disables automatic port extensions on the specified port.

Parameters

<cnun> Any existing channel number. If unspecified, value is set to 1

<n> PNA Port number to enable or disable for automatic port extensions.

<bool> ON (or 1) - Enable

OFF (or 0) - Disable

Examples

```
SENS:CORR:EXT:AUTO:PORT2 0  
sense2:correction:extension:auto:port4 on
```

Query Syntax SENSe<cnun>:CORRection:EXTension:AUTO:PORT<n>?

Return Type Boolean

Default All ports ON (enabled)

SENSe<cnun>:CORRection:EXTension:AUTO:RESet

(Write-only) Clears old port extension delay and loss data in preparation for acquiring new data. Send this command prior to sending a new series of [SENS:CORR:EXT:AUTO:MEAS](#). If acquiring both OPEN and SHORT standards, do not send this command between those acquisitions.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

Examples

```
SENS:CORR:EXT:AUTO:RES  
sense2:correction:extension:auto:reset
```

Query Syntax Not Applicable

Default Not Applicable

SENSe<cnum>:CORRection:EXTension:AUTO:STARt <value>

(Read-Write) Set the start frequency for custom user span. [Learn more about User Span](#).

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<value> User span start value. Must be within the frequency range of the active channel and less than the value set by SENS:CORR:EXT:AUTO:STOP.

Examples

```
SENS:CORR:EXT:AUTO:STAR 1E9  
sense2:correction:extension:auto:start 200e6
```

Query Syntax SENSe<cnum>:CORRection:EXTension:AUTO:STARt <value>?

Return Type Numeric

Default Start frequency of the current active channel.

SENSe<cnum>:CORRection:EXTension:AUTO:STOP <value>

(Read-Write) Set the stop frequency for custom user span. [Learn more about User Span.](#)

Parameters

- <cnm> Any existing channel number. If unspecified, value is set to 1
- <value> User span stop value. Must be within the frequency range of the active channel and greater than the value set by SENS:CORR:EXT:AUTO:START

Examples

```
SENS:CORR:EXT:AUTO:STOP 1E9  
sense2:correction:extension:auto:stop 200e6
```

Query Syntax SENSE<cnm>:CORRection:EXTension:AUTO:STOP <value>?

Return Type Numeric

Default Stop frequency of the current active channel.

SENSe<cnm>:CORRection:EXTension:PORT<pnum>:DISTance <value>

(Read-Write) Sets and returns the port extension delay in physical length (distance).

Parameters

- <cnm> Any existing channel number. If unspecified, value is set to 1.
- <pnum> Port Number that will receive the delay setting. If unspecified, value is set to 1.
- <value> Physical length of fixture of added transmission line. First specify units with [SENS:CORR:EXT:PORT:UNIT](#).

Examples

```
SENS:CORR:EXT:PORT1:DIST 12  
sense2:correction:extension:port2:distance .003
```

Query Syntax SENSe<cnm>:CORRection:EXTension:PORT<pnum>:DISTance?

Return Type Numeric

Default 0

SENSe<cnm>:CORRection:EXTension:PORT<pnum>:FREQuency<n> <value>

(Read-Write) Sets and returns the frequency for the Freq and Loss pair number and for the specified port number.

[Learn about Loss Compensation values.](#)

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <pnum> Port Number that will receive the freq/loss settings. If unspecified, value is set to 1.
- <n> Freq and Loss pair number. Choose from 1 or 2. If unspecified, value is set to 1.
- <value> Frequency value. Choose a frequency within the frequency span of the PNA.

Examples

```
SENS:CORR:EXT:PORT1:FREQ1 10E9  
sense2:correction:extension:port2:freq2 2E10
```

Query Syntax SENSE<cnum>:CORRection:EXTension:PORT<pnum>:FREQuency<n>?

Return Type Numeric

Default 1 GHz

SENSe<cnum>:CORRection:EXTension:PORT<pnum>:INCLude<n>[:STATE] <bool>

(Read-Write) Sets and returns the ON/OFF state for the Freq and Loss pair number and for the specified port number.

[Learn about Loss Compensation values.](#)

Note: This command affects ALL measurements on the specified channel.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <pnum> Port Number that will receive the Freq/Loss settings. If unspecified, value is set to 1.
- <n> Freq and Loss pair. Choose from 1 or 2. If unspecified, value is set to 1.
- <value> State of Freq and Loss values for port extension.

0 or OFF Specified Freq and Loss values are OFF

1 or ON Specified Freq and Loss values are ON

Examples

```
SENS:CORR:EXT:PORT:INCL 0  
sense2:correction:extension:port2:include2:state on
```

Query Syntax SENSE<cnum>:CORRection:EXTension:PORT<pnum>:INCLude[:STATe]?

Return Type Boolean

Default OFF

SENSE<cnum>:CORRection:EXTension:PORT<pnum>:LDC <value>

(Read-Write) Sets and returns the Port Loss at DC value for the specified port number.

[Learn about Loss Compensation values.](#)

Note: This command affects ALL measurements on the specified channel.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <pnum> Port number to receive Loss value. If unspecified, value is set to 1.
- <value> Loss in dB. Choose a value between -90 and 90

Examples

```
SENS:CORR:EXT:PORT:LDC 1.5  
sense2:correction:extension:port2:ldc .1
```

Query Syntax SENSE<cnum>:CORRection:EXTension:PORT<pnum>:LDC?

Return Type Numeric

Default 0

SENSE<cnum>:CORRection:EXTension:PORT<pnum>:LOSS<n> <value>

(Read-Write) Sets and returns the Loss value for the specified port number.

[Learn about Loss Compensation values.](#)

Note: This command affects ALL measurements on the specified channel.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <pnum> Port Number that will receive the Freq/Loss settings. If unspecified, value is set to 1.
- <n> Loss "Use" number. Choose from 1 or 2. If unspecified, value is set to 1.
- <value> Loss in dB. Choose a value between -90 and 90

Examples `SENS:CORR:EXT:PORT:LOSS1 1`

`sense2:correction:extension:port2:loss2 .1`

Query Syntax `SENSe<cnum>:CORRection:EXTension:PORT<pnum>:LOSS<n>?`

Return Type Numeric

Default 0

SENSe<cnum>:CORRection:EXTension:PORT<pnum>:MEDium <char>

(Read-Write) Sets and returns the media type of the added fixture or transmission line.

See also [SENS:CORR:EXT:PORT:SYSMedia](#)

Note: This command affects ALL measurements on the specified channel.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<pnum> Port Number for which media type is being set. If unspecified, value is set to 1.

<char> Medium type. Choose from:

- COAX
- WAVEguide

Examples `SENS:CORR:EXT:PORT:MED COAX`

`sense2:correction:extension:port2:medium waveguide`

Query Syntax `SENSe<cnum>:CORRection:EXTension:PORT<pnum>:MEDium?`

Return Type Character

Default COAX

SENSe<cnum>:CORRection:EXTension:PORT<pnum>:SYSMedia <bool>

(Read-Write) Sets and returns the state of coupling with the system Media type. [Learn more.](#)

Note: This command potentially affects ALL measurements on the PNA.

Parameters

- <num> Any existing channel number. If unspecified, value is set to 1
- <pnum> Port Number for which system Velocity Factor coupling is being set. If unspecified, value is set to 1.
- <bool> Coupling state. Choose from:
- **ON** (or 1) - Media type is coupled with the system setting.
 - **OFF** (or 0) - Media type is NOT coupled with the system setting.

Examples

```
SENS:CORR:EXT:PORT:SYSM 1  
sense2:correction:extension:port2:sysmedia off
```

Query Syntax SENSE<num>:CORRection:EXTension:PORT<pnum>:SYSMedia?

Return Type Boolean

Default 1 or ON (Coupled)

SENSe<num>:CORRection:EXTension:PORT<pnum>:SYSVelocity <bool>

(Read-Write) Sets and returns the state of coupling with the system Velocity Factor value. [Learn more.](#)

Note: This command potentially affects ALL measurements on the PNA.

Parameters

- <num> Any existing channel number. If unspecified, value is set to 1
- <pnum> Port Number for which system Velocity Factor coupling is being set. If unspecified, value is set to 1.
- <bool> Coupling state. Choose from:
- **ON** (or 1) - Velocity Factor is coupled with the system setting.
 - **OFF** (or 0) - Velocity Factor is NOT coupled with the system setting.

Examples

```
SENS:CORR:EXT:PORT:SYSV 1  
sense2:correction:extension:port2:sysvelocity off
```

Query Syntax SENSE<cnum>:CORRection:EXTension:PORT<pnum>:SYSVelocity?
Return Type Boolean
Default 1 or ON (Coupled)

SENSE<cnum>:CORRection:EXTension:PORT<pnum>[:TIME] <num>

(Read-Write) Sets the extension delay value in time at the specified port. Must also set [SENS:CORR:EXT ON](#).

Note: This command affects ALL measurements on the specified channel.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
<pnum> Port Number that will receive the extension. If unspecified, value is set to 1.
<num> The port extension in seconds; may include suffix. Choose a number between: -1E18 and 1E18

Examples

```
SENS:CORR:EXT:PORT 2MS  
sense2:correction:extension:port2 .00025
```

Query Syntax SENSE<cnum>:CORRection:EXTension:PORT<pnum> [:TIME]?
Return Type Numeric
Default 0

SENSE<cnum>:CORRection:EXTension:PORT:UNIT <char>

(Read-Write) Sets and returns the units for specifying port extension delay in physical length (distance).

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1.
<char> Units for delay in distance. Choose from:

- METer
- FEET
- INCH

Examples

```
SENS:CORR:EXT:PORT:UNIT MET  
sense2:correction:extension:port:unit inch
```

Query Syntax SENSE<cnum>:CORRection:EXTension:PORT:UNIT?

Return Type Character

Default METer

SENSe<cnum>:CORRection:EXTension:PORT<pnum>:VELFactor <value>

(Read-Write) Sets and returns the velocity factor of the fixture or added transmission line.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<pnum> Port Number for which velocity factor is being set. If unspecified, value is set to 1.

<value> Velocity Factor.

Set [SENS:CORR:EXT:PORT:SYSV](#) to use the system velocity factor.

Examples

```
SENS:CORR:EXT:PORT:VELF .6
```

```
sense2:correction:extension:port2:velfactor 1
```

Query Syntax SENSE<cnum>:CORRection:EXTension:PORT<pnum>:VELFactor?

Return Type Numeric

Default System Velocity Factor

SENSe<cnum>:CORRection:EXTension:PORT<pnum>:WGCutoff <value>

(Read-Write) Sets and returns the cutoff (minimum) frequency of the added waveguide fixture or transmission line.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<pnum> Port Number for which media type is being set. If unspecified, value is set to 1.

<value> Cutoff frequency in Hz.

This value is ignored when [SENS:CORR:EXT:PORT:MED](#) is set to **COAX** for the same port.

Examples

```
SENS:CORR:EXT:PORT:WGC 1e8
```

```
sense2:correction:extension:port2:wgcutoff 100Mhz
```

Query Syntax SENSE<cnum>:CORRection:EXTension:PORT<pnum>:WGCutoff?

Return Type Numeric

Default System Media Cutoff Frequency

SENSe<cnum>:CORRection:EXTension:RECeiver<Rnum>[:TIME] <num> OBSOLETE

(Read-Write) This command has NO replacement and no longer works.

Sets the extension value at the specified receiver. Must also set [SENS:CORR:EXT ON](#).

Note: Before using this command you must select a measurement using [CALC:PAR:SEL](#). You can select one measurement for each channel.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<Rnum> Number of the receiver that will receive the extension. If unspecified, value is set to 1

Choose from:

1 for Receiver A

2 for Receiver B

<num> The electrical length in seconds; may include suffix. Choose a number between:
-10 and **10**

Examples

```
SENS:CORR:EXT:REC 2MS  
sense2:correction:extension:receiver2:time .00025
```

Query Syntax SENSe<cnum>:CORRection:EXTension:RECeiver<Rnum>[:TIME]?

Return Type Numeric

Default 0

SENSe<cnum>:CORRection:EXTension[:STATe] <ON | OFF>

(Read-Write) Turns port extensions ON or OFF.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<ON | OFF> **ON** (or 1) - turns port extensions ON.

OFF (or 0) - turns port extensions is OFF.

Examples

```
SENS:CORR:EXT ON  
sense2:correction:extension:state off
```

Query Syntax SENSE<cnum>:CORRection:EXTension[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

Last Modified:

- 26-Aug-13 Another minor edit
- 11-Jan-2011 Minor edits
- 28-Sep-2009 Fixed Port:Frequency command syntax
- 6-Feb-2009 Added new commands (A.08.50)

Sense:Correction:Collect:Guided Commands

Performs and applies a SmartCal (Guided) calibration and other error correction features.

To perform a **Guided Calibration**, use ONLY [Sens:Corr:Coll:Guided](#) commands. See the "[Guided](#)" [example programs](#) for clarification.

SENSe:CORRection:COLLect:GUIDed:

ABORt

ACQuire

ADAPter

| **COUNt**

| **ZERO**

| **CREate?**

| **DELay**

| **DESCription**

| **PATHs**

CHANnel:MODE

CKIT

| **CATalog?**

| **PORT**

| **CATalog?**

| **[SElect]**

CONNector

| **CATalog?**

| **PORT**

| **[SElect]**

DESCription

DMATch

| **APPLy**

| **[IMMediate]**

| **PORTs?**

| **[INITiate]**

ETERms:LOAD

| **[CSET]**

INITiate

ISOLation

| **AVERage**

| **INCRement**

| **PATHs**

ITERations

| [COUNT?](#)

| [MINimum?](#)

| [RESet](#)

METHod

[PACQuire](#)

PATH

| [CMETHod](#)

| [TMETHod](#)

PREference

| [SLIDingload](#)

[PSENSor - More commands](#)

SAVE

| [CSET](#)

[SMC - More commands](#)

STEPS?

THRU

[VMC - More commands](#)

Click on a [blue](#) keyword to view the command details.

[Red](#) keywords are superseded commands.

See Also

[ECal Orientation commands](#)

[Examples](#) using these commands.

[Calibrating the PNA Using SCPI](#)

[Learn about Measurement Calibration](#)

[Synchronizing the PNA and Controller](#)

SENSe<ch>:CORRection:COLLEct:GUIDed:ABORt

(Write-only) Aborts the acquiring of a guided calibration that has been [INITIALIZED](#) but has not yet been concluded using the [SAVE](#) command. If at least one Cal standard has already been measured, and the [Calibration Window](#) is being displayed, this command also closes the Calibration Window and re-tiles the other measurement windows.

Parameters

<ch> Channel being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.

Examples

```
SENS:CORR:COLL:GUID:ABOR
sense2:correction:collect:guided:abort
```

Query Syntax Not Applicable

Default Not Applicable

SENSe:<ch>CORRection:COLLEct:GUIDed[:ACQuire] <std>[,sync]

(Write-only) Initiates the measurement of the specified calibration standard. Executing this command with an unnecessary standard has no affect.

The measured data is stored and used for subsequent calculations of error correction coefficients. All standards must be measured before a calibration can be completed. Any measurement can be repeated until the [SENS:CORR:COLL:GUID:SAVE](#) command is executed.

Query the user prompt description using [SENS:CORR:COLL:GUID:DESC?](#)

Query the required calibration steps using [SENS:CORR:COLL:GUID:STEP?](#)

Parameters

<ch> Channel being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.

<std> Choose from:STAN1, STAN2, STAN3, through STAN40

[sync] Optional argument. Choose from:

SYNChronous - blocks SCPI commands during standard measurement (default behavior).

ASYNchronous - does NOT block SCPI commands during standard measurement.

[Learn more about this argument](#)

Examples

```
SENS:CORR:COLL:GUID STAN1
sense2:correction:collect:guided:acquire stan1
```

Query Syntax Not Applicable

Default Not Applicable

SENSe<ch>:CORRection:COLLEct:GUIDed:ADAPter:CREate? <conn1>, <conn2>

(Read-only) Specifies the use of a THRU adapter to be used during the Guided Cal Unknown THRU and Adapter Removal Cal. Returns an adapter index <n> which is used to refer to the adapter in several related commands. [See Cal Thru Methods](#). While the choice of which end of the adapter is <conn1> and <conn2> is arbitrary, it is necessary to remember which will be used on each test port.

The settings for this command remain until Preset, or the command is sent using a different setting, or until the [ZERO](#) command is sent.

Parameters

- <ch> Channel being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.
- <conn1> Adapter port 1 connector type. Use [SENS:CORR:COLL:GUID:CONN:CAT?](#) to return a list of valid connector types.
- <conn2> Adapter port 2 connector type.

Examples [See example using this command.](#)

Return Type Numeric

Default Not Applicable

SENSe<ch>:CORRection:COLLEct:GUIDed:ADAPter:COUNt?

(Read-Only) Returns the number of THRU adapters that have been created for this calibration using [SENS:CORR:COLL:GUID:ADAP:CREate](#).

Parameters

- <ch> Channel being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.

Examples [See example using this command.](#)

Return Type Numeric

Default Not Applicable

SENSe:<ch>CORRection:COLLEct:GUIDed:ADAPter:COUNt:ZERO

(Write-only) Removes all adapters that have been defined for calibrations on the specified channel using [SENS:CORR:COLL:GUID:ADAP:CREate](#).

Parameters

<ch> Channel being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.

Examples

```
SENS:CORR:COLL:GUID:ADAP:COUNT:ZERO  
sense2:correction:collect:guided:adapter:count:zero
```

Query Syntax Not Applicable

Default Not Applicable

SENSe<ch>:CORRection:COLLEct:GUIDed:ADAPter<n>:DELay <coax>, [w phase, wdelay]

(Write-only) Specifies the adapter delay, and optionally waveguide delay and optional phase offset (degrees) of adapter <n>.

The settings for this command remain until Preset, or the command is sent using a different setting, or until the [ZERO](#) command is sent.

Parameters

<ch> Channel being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.

<n> Adapter index number that was returned from [SENS:CORR:COLL:GUID:ADAP:CREate?](#)

<coax> Delay value of coax adapter <n> in seconds. If the adapter has no coax connector, enter 0.

<wphase> Waveguide phase offset in degrees. If the adapter has no waveguide connector, do not enter a value.

<wdelay> Waveguide delay in seconds. If the adapter has no waveguide connector, do not enter a value.

Examples

[See example using this command.](#)

Default Not Applicable

SENSe<ch>:CORRection:COLLEct:GUIDed:ADAPter<n>:DESCRiption <string>

(Write-only) Specifies the adapter description for use as the guided cal connection prompts.

The settings for this command remain until Preset, or the command is sent using a different setting, or until the [ZERO](#) command is sent.

Parameters

- <ch> Channel being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.
- <n> Adapter index number that was returned from [SENS:CORR:COLL:GUID:ADAP:CREate?](#)
- <string> Adapter description.

Examples [See example using this command.](#)

Query Syntax Not Applicable

Default Not Applicable

SENSe<ch>:CORRection:COLLect:GUIDed:ADAPter<n>:PATHs <port pairs>

(Write-only) Specifies the port pairs for which the adapter will be used for a THRU connection.

For example, for a 3-port cal on channel 1 using ports 1,2,and 3), to use adapter 1 between the ports (1 to 2) and (1 to 3) the following command is used: SENS1:CORR:COLL:GUID:ADAP1:PATH 1,2,1,3.

The adapter must have the same DUT connectors as the ports that are already specified for these ports.

The settings for this command remain until Preset, or the command is sent using a different setting, or until the [ZERO](#) command is sent.

Parameters

- <ch> Channel being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.
- <n> Adapter index number that was returned from [SENS:CORR:COLL:GUID:ADAP:CREate?](#)
- <port pair> Ports for which the adapter will be used. The orientation is not critical, as the PNA will align the connector types as necessary. The minimum number of Thru connections required is the number of ports to calibrated -1.

Examples [See example using this command.](#)

Query Syntax Not Applicable

Default Not Applicable

SENSe:CORRection:COLLect:GUIDed:CHANnel:MODE <bool>

(Read-Write) Determines whether or not to honor the channel <ch> argument in guided calibration SCPI commands.

Parameters

<bool> **OFF (0)** Honor all <ch> arguments. This means the <ch> channel is calibrated regardless of which channel is currently active.

ON (1) Legacy behavior. Behavior is specified by the following table:

<ch> channel type	Active channel type	Behavior
Std or App	Std or App	
Std	Std	Active chan cal'd
Std	App	"Channel not found" error
App	Std	<ch> chan cal'd
App	App	<ch> chan cal'd

Learn about [Standard vs Application](#) channels.

Examples

```
SENSe:CORR:COLL:GUID:CHAN:MODE 0
```

```
sense:correction:collect:guided:channel:mode ON
```

Query Syntax SENSe:CORRection:COLLect:GUIDed:CHANnel:MODE?

Return Type Boolean

Default OFF - This is the default beginning with A.09.50

ON - Default before A.09.50

SENSe:CORRection:COLLect:GUIDed:CKIT:CATalog? <connector>

(Read-only) This command replaces [SENS:CORR:COLL:GUID:CKIT:PORT:CAT?](#)

Returns a comma-separated list of valid kits that use the specified connector type. This includes mechanical cal kits, applicable characterizations found within ECal modules currently connected to the PNA, **and all user characterizations stored in PNA disk memory**. For ECal modules, the returned list includes the serial numbers. [See ECal User Characterization commands](#).

Use items in the list to select the kit to be used with the [SENS:CORR:COLL:GUID:CKIT:PORT](#) and [SENSe<ch>:CORRection:COLLect:GUIDed:PSEnsor<pnum>:CKIT](#) commands.

Parameters

<conn> String. Connector type. Use [SENS:CORR:COLL:GUID:CONN:CAT?](#) to return a list of valid connector types.

Examples

```
SENSe:CORR:COLL:GUID:CKIT:CAT? "Type N (50) male"
```

Return Type String

Default Not Applicable

SENSe:CORRection:COLLect:GUIDed:CKIT:PORT<pnum>:CATalog? Superseded

(Read-only) This command is replaced by [SENSe:CORR:COLL:GUID:CKIT:CAT?](#)

Returns a comma-separated list of valid kits for the specified PNA port. In addition to mechanical calibration kits, this will include applicable characterizations found within ECal modules currently connected to the PNA.

Use items in the list to select the kit to be used with the [SENS:CORR:COLL:GUID:CKIT:PORT](#) command.

Note: Beginning with PNA Rev 9.1, the serial number is returned for ALL ECal modules that are connected with the connector type of the specified port. Previously, the returned list would include the serial numbers to distinguish the ECal modules only when two or more identical ECal models were connected to the PNA.

Parameters

<pnum> Any existing port number. If unspecified, value is set to 1

Examples

```
SENS:CORR:COLL:GUID:CKIT:PORT1:CAT?
```

```
'When "Type N (50) male" is specified for connector type,  
returns:
```

```
"85054D, 85032F"
```

```
'When two identical ECal modules are connected for the connector  
type,
```

```
'the return string includes serial numbers
"85092-60010 ECal 10675, 85092-60010 ECal 00758"
```

Return Type String

Default Not Applicable

SENSe<ch>:CORRection:COLLect:GUIDed:CKIT:PORT<pnum>[:SElect] <kit>

(Read-Write) Specifies the calibration kit (mechanical or ECal) for each port to be used during a guided calibration. An unused port does NOT need to have a specified Cal Kit.

1. Specify the connector type for the port with [SENS:CORR:COLL:GUID:CONN:PORT](#).
2. Query the valid available kits for the connector on each port with [SENS:CORR:COLL:GUID:CKIT:PORT:CAT?](#)
3. Specify the kit using this command.
4. Perform a query of this command. If the <kit> parameter was incorrectly entered, an error will be returned.

When using this command to specify the cal kit for the output of a VMC calibration mixer, specify port 3. If port 3 is already used for the output of the DUT mixer, then specify port 4. [Learn more](#).

Parameters

- <ch> Channel being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.
- <pnum> Any existing port number. If unspecified, value is set to 1
- <kit> Calibration kit to be used for the specified port. **Case-sensitive.**

When using an ECal module, include the characterization name in the <kit> string. Use [SENSe:CORR:COLL:GUID:CKIT:CAT?](#) to read the list of characterizations available in the module and in PNA disk memory.

If two or more identical ECal modules are connected to the PNA, the serial number must be included to distinguish the ECal modules.

Examples

```
'Note: All of the following examples specify port 1 only
' Mechanical Cal kit
SENS:CORR:COLL:GUID:CKIT:PORT1 '85055A'
' Standard ECal modules
SENS:CORR:COLL:GUID:CKIT:PORT1 "N4691-60004 ECal"
```

' Non-factory ECal characterizations are specified as follows:

```
SENS:CORR:COLL:GUID:CKIT:PORT1 "N4691-60004 User 1 ECal"
```

' When two or more ECal modules with the same model number are
' connected, also specify the serial number as follows:

```
SENS:CORR:COLL:GUID:CKIT:PORT1 "N4691-60004 ECal 01234"
```

' When Disk Memory ECal user characterizations are used,
' specify both the User char and the serial number as follows:

```
SENS:CORR:COLL:GUID:CKIT:PORT1 "N4691-60004 MyDskChar ECal  
01234"
```

Query Syntax SENSE:CORRection:COLLect:GUIDed:CKIT:PORT<pnum>[:SElect]?

Return Type String - If the <kit> parameter was incorrectly entered while writing, an error will be returned.

Default Not Applicable

SENSe:CORRection:COLLect:GUIDed:CONNector:CATalog?

(Read only) Returns a list of valid connectors based on the connector descriptions of the available cal kits. Use an item from the returned list to specify a connector for

[SENS:CORR:COLL:GUID:CONN:PORT](#)

Here are the more common connector types:

W-band waveguide	Type B	1.00 mm female
V-band waveguide	Type A (50) female	1.00 mm male
U-band waveguide	Type A (50) male	1.85 mm male
R-band waveguide	Type F (75) female	1.85 mm female
Q-band waveguide	Type F (75) male	2.92 mm female
K-band waveguide	Type N (75) female	2.92 mm male
P-band waveguide	Type N (75) male	APC 2.4 female
X-band waveguide	Type N (50) female	APC 2.4 male
7-16 female	Type N (50) male	APC 3.5 female
7-16 male		APC 3.5 male

Examples

```
SENS:CORR:COLL:GUID:CONN:CAT?
```

Returns:

```
Type N (50) female, Type N (50) male, APC 7 (50), 3.5 mm (50)
male, 3.5 mm (50) female, User Connector A
```

Return Type Comma separated string values

Default Not Applicable

SENSe<ch>:CORRection:COLLect:GUIDed:CONNector:PORT<pnum>[:SElect] <conn>

(Read-Write) Specifies a connector type for every port during the Guided Calibration procedure. Valid connector names are stored within calibration kits. Some cal kits may include both male and female connectors. Therefore, specifying connector gender may be required.

The PNA remembers previous Guided Cal settings. Therefore, for completeness, unused ports should be defined as "Not used". See [Guided Cal examples](#).

- A single port with a valid <conn> name indicates a 1-Port calibration will be performed.
- Two ports with valid <conn> names indicate either a 2-Port SOLT or [TRL](#) calibration will be performed depending on the standards definition found within the cal kit and the capability of the PNA.
- Three ports with valid <conn> names indicate a 3-Port calibration will be performed, and so forth.

Follow these steps to ensure port connectors are specified correctly:

1. Use [SENS:CORR:COLL:GUID:CONN:CAT?](#) to query available connectors before specifying the port connector.
2. Set a connector type for each port using this command.
3. Perform a query of this command. If the connector type was incorrectly entered, an error will be returned.
4. Specify the cal kit to use for each port with [SENS:CORR:COLL:GUID:CKIT:PORT](#)

Parameters

<ch> Channel being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.

<pnum> Any existing port number. If unspecified, value is set to 1.

<conn> String - DUT connector type to connect with PNA port <pnum>. **Case-sensitive.**

Examples

```
'Specifying a 2-port cal (1 & 2) on a 4-port PNA  
SENS:CORR:COLL:GUID:CONN:PORT1 'Type N (50) female'  
SENS:CORR:COLL:GUID:CONN:PORT2 'Type N (50) male'  
SENS:CORR:COLL:GUID:CONN:PORT3 'Not used'  
SENS:CORR:COLL:GUID:CONN:PORT4 'Not used'
```

Query Syntax SENSE:CORRection:COLLEct:GUIDed:CONNector:PORT<pnum>[:SElect]?

Return Type String

Default Not Applicable

SENSe<ch>:CORRection:COLLEct:GUIDed:DESCription? <step>

(Read-only) Returns the connection description for the specified calibration step.

Parameters

<ch> Channel being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.

<step> A number from 1 to the number of steps required to complete the calibration (Use [SENS:CORR:COLL:GUID:STEP?](#) to query the number of steps)

Examples

```
SENS:CORR:COLL:GUID:DESC? 10  
  
'Returns:  
Connect APC 7 Open to port3
```

Return Type String

Default Not Applicable

SENSe<ch>:CORRection:COLLEct:GUIDed:DMATch:APPLy[:IMMediate] [string]

(Write-only) Specifies a Delta Match Cal Set to be used for delta match correction.

If the User-performed Cal Set GUID or Name is not specified, then the Global Delta Match Cal Set is applied.

An error is returned if the **specified** Cal Set does not meet the following Delta Match criteria:

- Must have been performed using ECal or as a guided mechanical cal (not Unguided).
- Must have the same start freq, stop freq, and number of points as the channel being calibrated.
- Must calibrate the ports that are required by the TRL or Unknown Thru cal as indicated by [SENS:CORR:COLL:GUID:DMATch:APPLY:PORTs?](#).

The Global Delta Match Cal can ALWAYS be applied.

[Learn more about Delta match calibration.](#)

See example of a complete [Global Delta Match calibration.](#)

See example where [Delta Match is applied to a calibration.](#)

Parameters

- <ch> Channel being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.
- [string] Optional argument. GUID or Name of the User-performed Delta Match Cal Set to apply.
- If unspecified, the Global Delta Match Cal Set is applied.

Examples

```
SENS:CORR:COLL:GUID:DMAT:APPL  
  
sense:correction:collect:guided:dmatch:apply:immediate  
"{2B893E7A-971A-11d5-8D6C-00108334AE96}"  
  
SENS:CORR:COLL:GUID:DMAT:APPL "MyDMatchCalset"
```

Query Syntax Not Applicable

Default Not Applicable

[SENSe<ch>:CORRection:COLLEct:GUIDed:DMATch:APPLY:PORTs?](#)

(Read-only) Returns the port numbers for which delta match correction is required. 0 (zero) is returned if the Cal does NOT require Delta Match correction for one of the following reasons:

- The Cal does NOT involve Unknown THRU or TRL. You specify this using [SENS:CORR:COLL:GUID:METH <UNKN | TRL>](#).
- The Cal DOES involve Unknown THRU or TRL, but the delta match data can be calculated by the Unknown Thru or TRL Cal. [Learn how this is possible](#). However, you can force the Cal to use the Delta Match data from a Cal Set.

[Learn more about Delta match calibration.](#)

See example of a complete [Delta Match calibration](#).

Parameters

<ch> Channel being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.

Examples

```
SENS:CORR:COLL:GUID:APPL:PORT?
```

```
'Returns:  
1,2,3
```

Return Type Numeric

Default Not Applicable

SENSe<ch>:CORRection:COLLEct:GUIDed:DMATch[:INITiate] <conn>,<cKit>

(Write-only) Initiates a global delta match calibration.

[Learn more about Global Delta Match calibration.](#)

See example of a complete [Delta Match calibration](#).

Parameters

<ch> Channel being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.

<conn> **String**. Connector type for port 1. All other ports are set automatically.

<cKit> **String** Cal Kit for all ports. If incorrectly entered while writing, an error is returned.

Examples

```
SENS:CORR:COLL:GUID:DMAT APC 3.5 female,"85052B"
```

Query Syntax Not Applicable

Default Not Applicable

SENSe<ch>:CORRection:COLLEct:GUIDed:ETERms:LOAD[:CSET] <cset>,<calPort> [,csPort]

(Write-only) Loads 1-port error terms from a Cal Set into the current Guided Cal sequence. When the Cal steps are recomputed, connection steps are removed due to the loading of the error terms.

[See example of how to use this command.](#)

Parameters

- <ch> **Channel** being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.
- <cset> **String** Name of User Cal Set in which the error terms reside.
- <pnum> **Integer** Port number of the current cal to receive error terms.
- [csPort] **Integer** Optional argument. Port number associated with the error terms in the Cal Set. If unspecified, the same port number as <calPort> is used.

Examples [See example](#)

Query Syntax Not Applicable

Default Not Applicable

SENSe<ch>:CORRection:COLLEct:GUIDed:INITiate[:IMMEDIATE] [string][, bool][,char]

(Write-only) Initiates a guided calibration.

- The PNA determines the measurements needed to perform the calibration using the settings specified from the [SENS:CORR:COLL:GUID:CONN:PORT](#) and [SENS:CORR:COLL:GUID:CKIT:PORT](#) commands.
- After this command is executed, subsequent commands can be used to query the number of measurement steps, issue the acquisition commands, query the connection description strings, and subsequently complete a guided calibration. [See example calibration programs.](#)

Parameters

- <ch> **Channel** to be calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.
- [string] **Optional argument.** Cal Set name or GUID enclosed in quotes.
- If NOT specified, behavior depends on the [SENS:CORR:PREFeRence:CSET:SAVE](#) setting.
- If specified, choose an **existing** Cal Set, either by name or by GUID.

- By Cal Set name: include quotes.
- By Cal Set GUID in the form: "{GUID}"; including quotes and curly brackets.
- Query all Cal Set GUIDs with [SENS:CORR:CSET:CAT?](#)

An error is reported if the Cal Set is not found.

The Cal Set is either supplemented or overwritten depending on the method, connectors, and ports selected. [Learn more about Cal Sets.](#)

[bool] Optional argument. To set this argument, also set the first optional argument. See example below.

OFF (0) If Cal Set stimulus settings differ from the existing channel, do not change channel stimulus settings. The Cal Set is saved to the current setting of the [SENS:CORR:PREF:CSET:SAVE](#) command. This is the default setting if not specified.

ON (1) If Cal Set stimulus settings differ from the existing channel, change the channel stimulus settings to match the Cal Set settings.

[char] Optional argument. To set this argument, also set the first two optional arguments. See example below.

SYNchronous - blocks further SCPI commands while processing this command.. (default setting).

ASYNchronous - does NOT block further SCPI commands while processing this command.

[Learn more about this argument](#)

Examples

```
SENS:CORR:COLL:GUID:INIT
'set first optional argument
SENS:CORR:COLL:GUID:INIT "MyCalSet"
'set two optional arguments
SENS:CORR:COLL:GUID:INIT " ",1
'set all optional arguments
SENS:CORR:COLL:GUID:INIT "MyCalSet",1,ASYN
```

Query Syntax Not Applicable

Default Not Applicable

SENSe<ch>:CORRection:COLLEct:GUIDed:ISOLation:AVERAge:INCRement <num>

(Read-Write) Specifies amount to increment (increase) the channels averaging factor during measurement of isolation standards in a guided calibration.

Note: If the channel has averaging turned OFF and the value of <num> is greater than 1, averaging will be turned ON only during the isolation measurements and with the averaging factor equal to <num>.

Parameters

- <ch> Channel being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.
- <num> Amount to increment the averaging factor for the isolation measurement. The maximum averaging factor for the channel is 65536 (2^{16}).

Examples

```
'Measure isolation on all paths for the cal
SENS:CORR:COLL:GUID:ISOL ALL
'Remove the port pairs 1-to-2 and 1-to-3 from the list of paths on
which to measure isolation
sense:correction:collect:guided:isolation:paths REMOVE,1,2,1,3
```

Query Syntax SENSe<ch>:CORRection:COLLEct:GUIDed:ISOLation:AVERAge:INCRement?

Return Type Numeric

Default 8 - If this command is NOT sent, but [isolation is measured](#), then averaging will be turned ON with factor set to 8 during the isolation measurements.

SENSe<ch>:CORRection:COLLEct:GUIDed:ISOLation[:PATHs] <char>[,<p1a, p1b, p2a, p2b]

(Read-Write) Specifies the paths (port pairs) to make isolation measurements on during a guided calibration.

Parameters

- <ch> Channel being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.
- <char> **ALL** Measure isolation on all pairings of the ports that are to be calibrated.
- NONE** Do not measure isolation on any pairing of the ports to be calibrated. (Default behavior).
- ADD** Add one or more specific pairings of ports to the list of port pairings for which isolation will be measured.

REMOve Remove one or more specific pairings of ports from the list of port pairings for which isolation will be measured. If many paths are to be measured, it may be easier to first send ALL, then REMove and specify the paths to remove.

<p1a, p2a...> For use when <char> is **ADD** or **REMOve**.

Specify Port numbers in pairs:

- For 3-port calcs, specify up to 3 pairs.
- For 4-port calcs, specify up to 6 pairs.

p1a, p1b (Path1 - port A and port B)

p2a, p2b (Path2 - port A and port B)

p3a, p3b (Path3 - port A and port B)

Examples

```
'Measure isolation on all paths for the cal
SENS:CORR:COLL:GUID:ISOL ALL

'Remove the port pairs 1-to-2 and 1-to-3 from the list of paths
on which to measure isolation

sense:correction:collect:guided:isolation:paths REMOVE,1,2,1,3
```

Query Syntax SENSE<ch>:CORRection:COLLEct:GUIDed:ISOLation:PATHs?

Note: if isolation is not be measured on any of the paths, the query returns 0

Return Type Numeric

Default 0 - Isolation not measured on any paths.

SENSe<ch>:CORRection:COLLEct:GUIDed:ITERations:COUNT? <step>

(Read-only) Designed to be used for an iterative cal standard such as a sliding load, this command returns the number of iterative measurement acquisitions that has been made for the specified step.

Zero (0) is returned if the step has not yet been measured.

For most cal steps that have already been measured, this command returns 1.

Set [SENS:CORR:COLL:GUID:PREF:SLIDITER](#) to count acquisition steps.

Parameters

- <ch> Channel being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.
 - <step> Guided Cal step number for which the acquisition number will be returned.
- Use [SENS:CORR:COLL:GUID:STEP?](#) to query the number of steps in the calibration.

Examples

```
SENS:CORR:COLL:GUID:ITER:COUN? 4
'Example return:
5
See example program
```

Return Type Numeric

Default Not Applicable

SENSe<ch>:CORRection:COLLEct:GUIDed:ITERations:MINimum? <step>

(Read-only) Designed to be used for an iterative cal standard such as a sliding load, this command returns the minimum number of required iterative measurement acquisitions for the specified step.

For most connection steps this will return 1, but for an iterative cal standard such as a sliding load, it will return a number such as 5.

Set [SENS:CORR:COLL:GUID:PREF:SLIDITER](#) to count acquisition steps.

Parameters

- <ch> Channel being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.

<step> Guided Cal step number for which to return the number of iterative measurement acquisitions that have been made. Use [SENS:CORR:COLL:GUID:STEP?](#) to query the number of steps in the calibration.

Examples

```
SENS:CORR:COLL:GUID:ITER:MIN? 4
```

```
'Example return:
```

```
5
```

```
See example program
```

Return Type Numeric

Default Not Applicable

SENSe<ch>:CORRection:COLLEct:GUIDed:ITERations:RESet <step>

(Write-only) Resets the specified guided cal connection step as unmeasured. This clears all previous measurements made for that step.

Parameters

<ch> Channel being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.

<step> Guided Cal step number to reset. Use [SENS:CORR:COLL:GUID:STEP?](#) to query the number of steps in the calibration.

Examples

```
SENS:CORR:COLL:GUID:ITER:RESet? 4
```

```
See example program
```

Return Type Not Applicable

Default Not Applicable

SENSe:CORRection:COLLEct:GUIDed:METhod <char> ~~Superseded~~

This command is replaced with [SENS:CORR:COLL:GUID:PATH:CMETHOD](#) and [SENS:CORR:COLL:GUID:PATH:TMETHOD](#).

(Read-Write) Selects from one of several algorithms available for performing the THRU portion of a guided calibration. [Learn more about THRU methods.](#)

Parameters

<char> **DEFAULT** - Informs guided calibrations to use the default algorithm when computing the number of needed standards acquisition steps. (default selection if omitted.)

ADAP - Use the adapter removal algorithm

FLUSH - Use with insertable devices.

UNKN - Use the Unknown THRU algorithm with calibrations for non-insertable devices.

DEFined - Use the THRU definition that you stored in the cal kit file, or ECal module.

TRL - Select TRL Cal Type for guided cals. Valid for "TRL ready" Cal Kits with properly assigned TRL cal classes.

SOLT - Select SOLT Cal Type for guided cals. Valid for any kit with properly assigned SOLT cal classes.

Examples

```
SENS:CORR:COLL:GUID:METH ADAP  
sense:correction:collect:guided:method unkn
```

Query Syntax SENSE:CORRection:COLLect:GUIDed:METHod?

Return Type Character

Default DEFAULT

SENSe<ch>:CORRection:COLLect:GUIDed:PACQuire <std>

(Write-only) Show the [Cal Window](#), and optionally one or more other specific windows before acquiring a Cal standard. This command will cause the Cal Window to display the specific measurements that are to be made for that particular Cal standard.

Parameters

- <ch> Channel being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.
- <std> Choose from:STAN1, STAN2, STAN3, through STAN40.

Examples

```
SENS:CORR:COLL:GUID:PACquire STAN2  
sense:correction:collect:guided:pacquire STAN5
```

[See an example that uses this command.](#)

Query Syntax Not Applicable

Default Not Applicable

**SENSe<ch>:CORRection:COLLEct:GUIDed:PATH:CMETHod
<port1>,<port2>,<caltype1[,caltype2]>**

Note: This command replaces [SENS:CORR:COLL:GUID:METH.](#)

(Read-Write) Specifies the calibration method for each port pair.

Note: Sending this command will overwrite the PNAs SmartCal determinations for the most accurate cal method for your connector settings and Cal Kits. Send this command ONLY if you have a deliberate reason for overwriting the SmartCal logic. You can send the query form of this command to learn the cal method determined by SmartCal.

See [Thru Pairs Sequence](#) to learn how to send this and other Thru commands.

After sending this command, send the query form to be sure that the command was accepted. If not, then the chosen Cal method is not compatible with the specified Thru method. For example, if the specified Thru method is Unknown Thru, an attempt to set Enhanced Response Cal should be rejected.

[Learn more about Thru Methods.](#)

Parameters

- <ch> Channel being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.
- <port1> First port of the pair to be calibrated.

<port2> Second port of the pair to be calibrated.
<caltype1[caltype2]> (String) Cal type for the port pair, enclosed in a single pair of quotes. NOT case-sensitive.

caltype1 Choose from:

- TRL
- SOLT
- QSOLTN
- EnhRespN

For the last two arguments, replace **N** with the port to be used as the source port, which **MUST** be one of the port pair.

caltype2 Optional argument. Use only when performing an adapter removal cal on the pair. This argument specifies the Cal type on the second port. Caltype1 then specifies the Cal type of the first port.

Choose from the same arguments as caltype1.

Examples

```
SENS:CORR:COLL:GUID:PATH:CMETHOD 2,3,"QSOLT2"
```

```
sense:correction:collect:guided:path:cmethod 2,3,"solt,trl"
```

Query Syntax

```
SENSe<ch>:CORRection:COLLect:GUIDed:PATH:CMETHOD?  
<port1>,<port2>
```

If only one caltype is returned then its NOT adapter removal.

Return Type

String

Default

The most accurate Cal method for the current cal.

**SENSe<ch>:CORRection:COLLect:GUIDed:PATH:TMETHOD
<port1>,<port2>,<thruType1[,thruType2]>**

Note: This command replaces [SENS:CORR:COLL:GUID:METH.](#)

(Read-Write) Specifies the calibration **THRU** method for each port pair.

Note: Sending this command will overwrite the PNAs SmartCal determination for the thru method. Send this command **ONLY** if you have a deliberate reason for overwriting the SmartCal logic. You can send the query form of this command to learn the THRU method determined by SmartCal.

See [Thru Pairs Sequence](#) to learn how to send this and other Thru commands.

[Learn more about Thru methods.](#)

Parameters

- <ch> Channel being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.
- <port1> First port of the port pair to be calibrated.
- <port2> Second port of the port pair to be calibrated.
- <thruType1[,thruType2]> (String) Thru methods for port pair, enclosed in a single pair of quotes. NOT case-sensitive.

thruType1 Calibration Thru method. Choose from:

- **Defined Thru** Measures a Thru for which there is a stored definition in the Cal kit of the lowest-numbered port of the pair. For example, if the port pair is 1,2, then the cal kit for port 1 MUST contain a Defined Thru.
- **Zero Thru** Measures a Zero length Thru, also known as Flush-Thru.
- **Undefined Thru** (Also known as Unknown Thru) A Thru type for which there is NOT a stored definition in the Cal Kit. Valid ONLY for SOLT cal type.
- **Undefined Thru using a Defined Thru** (ECal modules ONLY) Measures the internal Thru as an Unknown Thru.

thruType2 Optional argument. Use ONLY when Adapter Removal Cal is specified for the pair using [SENS:CORR:COLL:GUID:PATH:CMETHOD](#). When specifying ThruType2, this is the only valid argument: "**Defined Thru, Defined Thru**"

Examples

```
SENS:CORR:COLL:GUID:PATH:TMETHOD 2,3,"Zero Thru"  
sense:correction:collect:guided:path:tmethod 2,3,"Defined  
Thru,Defined Thru"
```

Query Syntax

```
SENSe<ch>:CORRection:COLLect:GUIDed:PATH:TMETHOD?  
<port1>,<port2>
```

Always returns two parts:

If the second part of the string is empty, adapter removal is NOT being performed.

If the string is "Defined Thru, Defined Thru", adapter removal IS being

performed.

Return Type String

Default The most accurate Thru method for the current cal.

SENSe<ch>:CORRection:COLLect:GUIDed:PREFErence:SLIDingload <char>

(Read-Write) Specifies the behavior for guided cal steps that involve a sliding load in a cal that is about to be performed. Send this command BEFORE sending the Guided INIT command.

Although the term 'Preference' is used in the command, this is NOT a PNA preference. This setting does NOT survive instrument preset or reboot. It remains ONLY for the duration of the Guided Cal.

Parameters

<char> Behavior when measurements of sliding load are acquired. Choose from:

DIALog - The Sliding load dialog box appears when the acquire command is received for a sliding load step. All slide positions are measured (with a user-interface prompt) from a single invocation of the acquire command.

ITERate - Each invocation of the acquire command for a sliding load step measures a single slide position and increments the slide position counter. No Move Sliding Load prompt is presented on the PNA screen.

Examples

```
SENS:CORR:COLL:GUID:PREF:SLID ITER
```

[See example program](#)

Query Syntax SENSe<ch>:CORRection:COLLect:GUIDed:PREFErence:SLIDingload?

Return Type Character

Default DIALog

SENSe<ch>:CORRection:COLLect:GUIDed:SAVE[:IMMEDIATE] [bool]

(Write-only) Completes the guided cal by computing the error correction terms, turning Correction ON, and saving the calibration to a Cal Set. If all of the required standards have not been measured, the calibration will not complete properly.

Parameters

<ch> Channel being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.

[bool] Optional argument. If unspecified, the default behavior is the current PNA preference setting of [SENSe:CORRection:PREFeRence:CSET:SAVE](#)

OFF (0) Save cal data ONLY to a Cal Register.

ON (1) Save cal data to a Cal Register and a User Cal Set. The filename is automatically generated.

- For [application channels](#) (Gain Compression, SMC/VMC, Noise Figure, IMD and so forth), this command saves ONLY to a Cal Register. Use [SENS:CORR:CSET:COPY](#) to copy the cal register to a named calset.
- For a [Calibrate All Channels](#) session, this argument is ignored. Instead, use [SYST:CAL:ALL:CSET:PREFix](#).

Examples

```
SENS:CORR:COLL:GUID:SAVE  
sense2:correction:collect:guided:save:immediate 0
```

Query Syntax Not Applicable

Default Not Applicable

SENSe<ch>:CORRection:COLLect:GUIDed:SAVE:CSET <cal set name or guid>

(Write-only) Completes the guided cal by computing the error correction terms, turning Correction ON, and saving the calibration to an existing, specified Cal Set. This command performs the same function as [SENSe:CORRection:COLLect:GUIDed:SAVE](#), except this command allows the name or GUID of the Cal Set to be specified.

Note: This command is NOT supported for [application channels](#) (Gain Compression, SMC/VMC, Noise Figure, IMD and so forth). Use [SENS:CORR:COLL:GUID:SAVE](#) and save to a cal register. You can then use [SENS:CORR:CSET:COPY](#) to copy the cal register to a named calset.

- Use this command instead of specifying the optional name or GUID argument in [SENS:CORR:COLL:GUID:INIT](#).
- Use [SENS:CORRection:CSET](#) commands to get names or GUIDs of existing Cal Sets.

- The cal data is also saved to the channel Cal Register.
- If all of the required standards have not been measured, the calibration will not complete properly.

For Calibrate All Channels

When this command is used during a Cal All session, the <cal set name> argument sets the User Cal Set prefix. All generated Cal Sets will be preceded with this string name.

- Cal Set prefix can also be set using [SYST:CAL:ALL:CSET:PREFIX](#). When the Cal Set prefix has already been set with [SYST:CAL:ALL:CSET:PREFIX](#), this command overwrites it.
- When <cal set name> is an empty string, a User Cal Set will not be saved. Only Cal Registers will be saved.

Parameters

- <ch> Channel being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.
- <cal set name or guid> **String** - Name or GUID of an existing Cal Set to be overwritten. If specifying a GUID, curly brackets must be included.

See [Calibrate All Channels](#) note (above).

Examples

```
SENS:CORR:COLL:GUID:SAVE:CSET "{2B893E7A-971A-11d5-8D6C-00108334AE96}"
```

```
sense:correction:collect:guided:save:cset "MyCalSet"
```

Query Syntax Not Applicable

Default Not Applicable

SENSe<ch>:CORRection:COLLEct:GUIDed:STEPs?

(Read-only) Returns the number of measurement steps required to complete the current guided calibration. This command is sent after the [SENS:CORR:COLL:GUID:INIT](#), [SENS:CORR:COLL:GUID:CONN:PORT](#) and [SENS:CORR:COLL:GUID:CKIT:PORT](#) commands.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.

Examples

```
SENS:CORR:COLL:GUID:STEP?  
sense2:correction:collect:guided:steps?
```

Return Type Numeric

Default Not Applicable

SENSe<ch>:CORRection:COLLEct:GUIDed:THRU:PORTs <t1a, t1b, t2a, t2b, t3a, t3b...>

(Read-Write) For calibrating more than 2-ports ONLY. Specifies the port pairs for the Thru connections of the calibration. Send the query form of this command to learn the Thru pairs determined by SmartCal.

Note: Sending this command will overwrite the PNAs SmartCal determinations for the thru ports. Send this command ONLY if you have a deliberate reason for overwriting the SmartCal logic.

See [Thru Pairs Sequence](#) to learn how to send this and other Thru commands.

Parameters

- <ch> Channel being calibrated, depending on the [CHAN:MODE](#) setting. If unspecified, value is set to 1.
- <t1a,t1b...> Always specify port numbers in pairs: For example: 1,2 or 1,2,1,3
- For 3-port cals, specify two or three pairs.
 - For 4-port cals, specify from three up to six pairs.

Examples

```
SENS:CORR:COLL:GUID:THRU:PORT 1,2,1,3,1,4 '4-port measurement  
sense:correction:collect:guided:thru:ports 1,2,2,3 '3-port  
measurement
```

Query Syntax SENSe<ch>:CORRection:COLLEct:GUIDed:THRU:PORTs?

Return Type Numeric

Default Port pairings that were used in the previous cal.

THRU Pairs sequence

The SmartCal logic always determines the best calibration based on your specified connectors and ports.

The following three commands overwrite the SmartCal logic. Send these commands ONLY if you have a deliberate reason for overwriting the SmartCal logic.

- [SENS:CORR:COLL:GUID:THRU:PORTS](#) <p1, p2>
- [SENS:CORR:COLL:GUID:PATH:TMET](#) <p1,p2, thrutype>
- [SENS:CORR:COLL:GUID:PATH:CMET](#) <p1,p2, calmethod>

When sending one or more of these commands, they must be sent in the following sequence with the other commands listed here.

Note: The **GUID:INIT** command is sent before and after these commands.

1. [SENS:CORR:COLL:GUID:CONN:PORT\(N\)](#)
 2. [SENS:CORR:COLL:GUID:CKIT:PORT\(N\)](#)
 3. [SENS:CORR:COLL:GUID:INIT](#)
 4. [SENS:CORR:COLL:GUID:THRU:PORTS](#) <P1, P2>
 5. [SENS:CORR:COLL:GUID:PATH:TMET](#) <P1,P2, THRUTYPE>
 6. [SENS:CORR:COLL:GUID:PATH:CMET](#) <P1,P2, CALMETHOD>
 7. [SENS:CORR:COLL:GUID:PATH:CMET?](#) <P1,P2> (RECOMMENDED)
 8. [SENS:CORR:COLL:GUID:INIT](#)
-

Last modified:

29-Jan-2014	Added notes to save for applications
19-Dec-2013	Modified TMethod for cal kit to use
5-Aug-2013	Edit PACQ to write only
11-Jun-2013	Fixed typo
4-Dec-2012	Added sliding load support
3-Dec-2012	Added THRU sequence
30-Nov-2012	Minor changes to Thru ports (SW)
1-Nov-2012	Added Calset name to DM apply (BH)
17-Apr-2012	Edit Init command for Cal All
28-Mar-2012	Edit Cset:Save command for Cal All
6-Dec-2011	Changed default for channel:mode command
5-Apr-2011	Modified ECal Cal Kit examples
11-Jan-2011	Several edits
29-Nov-2010	Moved Psensor commands to their own page (9.33)
22-Oct-2010	Linked Save
30-Jun-2010	Added chan:mode and Guided power cal (9.30)
16-Nov-2009	Added Note to GUID:CKIT:PORT<num>:CAT?
24-Aug-2009	Added ckitCat (9.0)
17-Jul-2009	Fixed order in which CMET and TMET are set.
27-Mar-2009	Fixed Isolation:Incr query
24-Feb-2009	Replaced True/False
13-Feb-2009	Added sync to ACQUIRE

26-Sep-2008	Added link to apply Delta Match example
10-Mar-2008	Added Abort command (A.07.50.27)
1-Nov-2007	Added PAcquire command
14-Apr-2007	Added Cal Set by name and isolation commands
8-Mar-2007	Added CMethod and TMethod
23-Oct-2006	Fixed wording in Conn:Port:Sel

Sense:Correction:IMD Commands

Controls an IMD and IMDx calibration. These commands supplement the [Guided Cal commands](#).

SENSe:CORRection:IMD

| CALibration

| FREQuencies

| METHod

| LO

| PCAL

| MPProduct

| POWer

| SENSor

| CKIT

| CONNector

| SORDer:INCLude

Click on a [blue](#) keyword to view the command details.

Other IMD (Opt 087) commands

- [CALC:CUSTom:DEFine](#) - creates an IMD measurement.
- [Swept IMD](#)
- [IMSpectrum](#)

See Also

- **Example** - [Create and Cal an IMD measurement](#)
 - [Learn about IMD Calibration](#)
 - [Learn about Measurement Class](#)
 - [Synchronizing the PNA and Controller](#)
 - [SCPI Command Tree](#)
-

SENSe<cnum>:CORRection:IMD:CALibration:FREQUencies <char>

(Read-Write) Sets and returns the frequencies at which an IMD source power cal is performed.

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1.

<char> Choose from:

CENTer - Perform source power calibration at only the center frequencies, midway between the main tones.

ALL - Perform source power calibration at all main tone frequencies.

Examples

```
SENS:CORR:IMD:CAL:FREQ ALL
```

```
sense2:correction:imd:calibration:frequencies center
```

Query Syntax SENSe<cnum>:CORRection:IMD:CALibration:FREQUencies?

Return Type Character

Default CENTer

SENSe<cnum>:CORRection:IMD:CALibration:METHod <char>

(Read-Write) Sets and returns the method by which the match-correction portion of an IMD calibration is performed. [Learn more.](#)

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1.

<char> Choose from:

MATCh - Performs a full 2-port cal for full match-correction.

RESPonse - Performs only a response (normalization) cal instead of a full 2-port cal.

Examples

```
SENS:CORR:IMD:CAL:METH MATC
```

```
sense2:correction:imd:calibration:method response
```

Query Syntax SENSe<cnum>:CORRection:IMD:CALibration:METHod?

Return Type Character

Default MATCh

SENSe<cnum>:CORRection:IMD:LO<n>:PCAL[:STATe] <bool>

(Read-Write) Sets and returns whether or not the LO power cal step is included in the cal steps when an IMDX cal is performed. [Learn more.](#)

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1.

<n> LO Stage. Choose 1.

<char> LO Power Cal state. Choose from:

0 or **OFF** - Skips over the LO Power Cal when calibrating.

1 or **ON** - Includes a step for LO Power Cal when calibrating

Examples

```
SENS:CORR:IMD:LO1:PCAL 0
```

```
sense2:correction:imd:lo1:pcal:state on
```

Query Syntax SENSE<cnum>:CORRection:IMD:LO<n>:PCAL[:STATe]?

Return Type Boolean

Default PNA Rev. 9.1 and above: **0** or **OFF**

Before Rev 9.1: **1** or **ON**

SENSe<cnum>:CORRection:IMD:MPROduct <num>

(Read-Write) Sets and returns the maximum intermod product frequencies to be calibrated. All lower product frequencies are also calibrated.

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1.

<char> Maximum IM products to calibrate. Choose from:

2 - second order products

3 - third order products

5 - fifth order products

7 - seventh order products

9 - ninth order products

Examples

```
SENS:CORR:IMD:MPR 5
```

```
sense2:correction:imd:mproduct 9
```

Query Syntax SENSE<num>::CORRection:IMD:MPROduct?
Return Type Numeric
Default 3

SENSe<num>:CORRection:IMD:POWer <num>

(Read-Write) Sets and returns the power level at which to perform the source power calibration using a power sensor.

Parameters

<num> Any existing channel number; if unspecified, value is set to 1.
<char> Power level. Choose a value between the min and max power level of the PNA.

Examples

```
SENS:CORR:IMD:POW -5  
sense2:correction:imd:power 5
```

Query Syntax SENSE<num>::CORRection:IMD:POWer?
Return Type Numeric
Default 0

SENSe<num>:CORRection:IMD:SENSor:CKIT <string>

(Read-Write) Sets and returns the cal kit to be used for calibrating at the port 1 reference plane when the power sensor connector is different from the DUT port 1. This effectively removes the effects of an adapter that is used to connect the power sensor.

Parameters

<num> Any existing channel number; if unspecified, value is set to 1.
<string> Cal Kit enclosed in quotes. First set the DUT connector for port 1 and the connector of the power sensor. Then use [SENS:CORR:COLL:GUID:CKIT:PORT1:CAT?](#) to return a list of valid cal kits.

Examples

```
SENS:CORR:IMD:SENS:CKIT "85052B"
```

Query Syntax SENSE<num>::CORRection:IMD:SENSor:CKIT?
Return Type String
Default Depends on the specified connectors

SENSe<cnum>:CORRection:IMD:SENSor:CONNector <string>

(Read-Write) Sets and returns the power sensor connector type which is used to perform the Source Power Cal portion of an IMD calibration.

Parameters

- <cnum> Any existing channel number; if unspecified, value is set to 1.
- <string> Power sensor connector type. Use [SENS:CORR:COLL:GUID:CONN:CAT?](#) to return a list of valid connector types.

Select "Ignored" to NOT compensate for the adapter.

Examples

```
SENS:CORR:IMD:SENS:CONN "APC 3.5 male"  
sense2:correction:imd:sensor:connector "Ignored"
```

Query Syntax SENSe<cnum>::CORRection:IMD:SENSor:CONNector?

Return Type String

Default "Ignored"

SENSe<cnum>:CORRection:IMD:SORDer:INCLude <bool>

(Read-Write) Sets and returns whether to include the second order products in the calibration. These frequencies of these products can be far from the main tones.

Parameters

- <cnum> Any existing channel number; if unspecified, value is set to 1.
- <bool> Choose from:

ON (or **1**) - Include 2nd order products

OFF (or **0**) - Do NOT include 2nd order products

Examples

```
SENS:CORR:IMD:SORD:INCL ON
```

Query Syntax SENSe<cnum>::CORRection:IMD:SORDer:INCLude?

Return Type Boolean

Default OFF or 0

Last Modified:

11-Jan-2011 Minor edit
27-Mar-2009 Edited for IMDx
18-Sep-2008 MX New topic

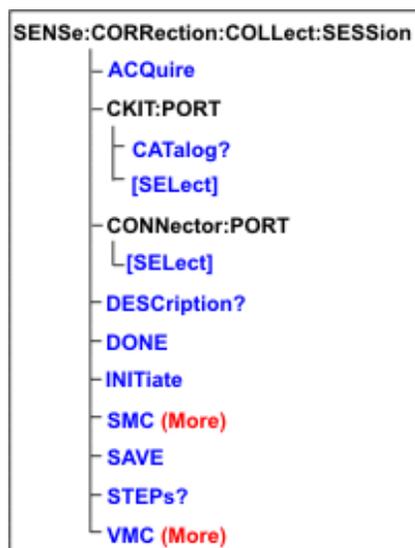
Sense:Correction:Collect:Session Commands - **Superseded**

Note: These commands are replaced with [Sense:Corr:Guided](#) commands.

The commands in this topic are common to perform both SMC and VMC calibrations.

A **calibration session** is a term used to describe an instance of a SMC or VMC calibration. The session number is chosen in the [SENS:CORR:COLL:SESS:INITiate](#) command. All other commands refer to that session number. For more commands, see [SESS:SMC](#) and [SESS:VMC](#).

Commands to read ([STEP?](#)) and describe ([DESC?](#)) each step are provided to facilitate a remote user interface.



Click on a [blue](#) keyword to view the command details.

See Also

- Learn about [SMC](#) and [VMC](#) calibrations
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

SENSe<ch>:CORRection:COLLect:SESSion<n>:ACQuire <step>[,sync]

(Write only) Acquire a calibration measurement.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
- <n> Session number of the calibration.
- [Learn about Cal sessions.](#)
- <step> Step number to acquire. Use [SENS:CORR:COLL:SESS:STEPS?](#) to find the number of steps required for the calibration.
- [sync] Optional argument. Choose from:
- SYNChronous** - blocks SCPI commands during standard measurement (default behavior).
- ASYNchronous** - does NOT block SCPI commands during standard measurement.
- [Learn more about this argument](#)

Examples

```
SENSe2:CORR:COLL:SESS6:ACQ 5,ASYN;*OPC?
```

Query Syntax Not Applicable

Default Not Applicable

SENSe<ch>:CORRection:COLLEct:SESSion<n>:CKIT:PORT<p>:CATalog?

(Read only) Returns a list of cal kits that are compatible with the connector on port <p>. The port connector type is set with [SENS:CORR:COLL:SESS:CONN:PORT](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
- <n> Session number of the calibration.
- [Learn about Cal sessions.](#)
- <p> PNA port number connector to query for compatible cal kits.
- For VMC, output port of the calibration mixer, specify 3 unless already used for the output of the mixer. Otherwise, specify 4.

Examples

```
SENS2:CORR:COLL:SESS6:CKIT:PORT2:CAT?
```

Return Type Comma separated string values

Default Not Applicable

SENSe<ch>:CORRection:COLLEct:SESSion<n>:CKIT:PORT<p>[:SELEct] <calkit>

(Read-Write) Set or return the Cal Kit for the specified port. Use [SENS:CORR:COLL:SESS:CKIT:PORT:CAT?](#) to list compatible Cal Kits.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<n> Session number of the calibration.

[Learn about Cal sessions.](#)

<p> PNA port number connector for which a cal kit is to be specified.

For VMC, output port of the calibration mixer, specify 3 unless already used for the output of the mixer. Otherwise, specify 4.

<calkit> Cal Kit Name

Examples

```
SENS:CORR:COLL:SESS:CKIT:PORT:SEL 85091A
SENS2:CORR:COLL:SESS6:CKIT:PORT2:SEL?
```

Query Syntax SENS<ch>:CORR:COLL:SESS<n>:CKIT:PORT<p>[:SEL] ?

Return Type String

Default Not Applicable

SENSe<ch>:CORRection:COLLEct:SESSion<n>:CONNector:PORT<p>[:SELEct] <conn>

(Read-Write) Set the connector type and sex for the specified port number. Catalog valid connector types using [SENS:CORR:COLL:GUID:CONN:CAT?](#)

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<n> Session number of the calibration.

[Learn about Cal sessions.](#)

<p> PNA port number connector for which to specify a connector type.

<conn> Name of the connector type

Examples

```
SENS2:CORR:COLL:SESS6:CONN:PORT1:SEL "N Type"
```

Query Syntax SENSe<ch>:CORR:COLL:SESS<n>:CONN:PORT<p>[:SELEct]?

Return Type String

Default Not Applicable

SENSe<ch>:CORRection:COLLEct:SESSion<n>:DESCription? <step>

(Read-only) Returns the connection prompt for the step. List the number of steps in the calibration using [SENS:CORR:COLL:SESS:STEPS?](#).

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<n> Session number of the calibration.

[Learn about Cal sessions.](#)

<step> Step number

Examples `SENS1:CORR:COLL:SESS6:DESC?3`

Return Type Numeric

Default Not Applicable

SENSe<ch>:CORRection:COLLEct:SESSion<n>:DONE

(Write only) Ends the calibration session. First use [SAVE?](#) to calculate error terms and save the CalSet.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<n> Session number of the calibration.

[Learn about Cal sessions.](#)

Examples `SENS1:CORR:COLL:SESS6:SAVE?`

`SENS1:CORR:COLL:SESS6:DONE`

Query Syntax Not Applicable

Default Not Applicable

SENSe<ch>:CORRection:COLLEct:SESSion<n>:INITiate <string>

(Write only) Initiates an SMC or VMC calibration session. Use the session number for subsequent SMC or VMC commands.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Session number. Choose from 1 to the [max number of channels](#).. If the session number already exists it will be terminated and a new session initiated.
- <string> Name of the calibration. Choose from:
 - "VMC" or "VectorMixerCal.VMCType"
 - "SMC" or "ScalarMixerCal.SMCType"

Examples

```
SENS1:CORR:COLL:SESS6:INITiate "VectorMixerCal.VMCType"
```

Query Syntax Not Applicable

Default Not Applicable

SENSe<ch>:CORRection:COLLEct:SESSion<n>:SAVE?

(Read only) Finish the SMC or VMC calibration, compute error terms, populate and save the Cal Set, and return the GUID of the Cal Set.

Note: The destination (Cal Register or User Cal Set) is determined by the setting of the [SENS:CORR:PREFeRence:CSET:SAVE](#) command.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Session number of the calibration.

[Learn about Cal sessions.](#)

Examples

```
SENS1:CORR:COLL:SESS6:SAVE?
```

Return Type String specifying the GUID of the Cal Set produced by this session.

Default Not Applicable

SENSe<ch>:CORRection:COLLEct:SESSion<n>:STEPs?

(Write-Read) Returns the number of steps required by the Calibration.

To ensure this query always completes successfully, first send the write command:
SENS:CORR:COLL:SESS:STEP without the <n> argument, then send the query.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Session number of the calibration.

[Learn about Cal sessions.](#)

Examples

```
SENS1:CORR:COLL:SESS6:STEPs?
```

Return Type Numeric

Default Not Applicable

Last Modified:

- 9-Aug-2013 Edit STEPs Read-Write
- 4-Jun-2009 Edits to DONE for clarification.
- 9-Apr-2009 Fixed commands for mixer port numbers.
- 13-Feb-2009 Added sync to ACQUIRE
- 5-Aug-2008 Edited sessions descriptions
- 13-Aug-2007 New image and detail for <port> argument.

Sense:Correction:Collect:Session:SMC Commands - **Superseded**

Note: Beginning with PNA release A..09.33, these commands (commonly known as "Session" commands) were replaced with [Sens:Corr:Coll:Guid:SMC](#) commands .

Performs scalar (SMC) calibration on a frequency converting device.

SENSE:CORRectionCOLLect:SESSion:SMC

| **ECAL**

| **CHARacterza**

| **FSIMulator**

| **NETWork**

| **FILEname**

| **MODE**

| **IMPort**

| **PHASe**

| **DELay**

| **METHod**

| **MIXer**

| **PWRCal**

| **SEParate**

| **SRCPort**

| **TWOPort**

| **ECAL**

| **ORientation**

| **[STATe]**

| **PORTmap**

| **METHod**

| **OMITisolat**

| **OPTion**

Click on a [blue](#) keyword to view the command details.

Red keywords are obsolete.

See Also

- [Example Programs](#)
- [Learn about SMC Calibrations](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

NOTE: To configure a power meter and sensor see [SOURCE:POWer:](#) commands.

SENSe<ch>:CORRection:COLLect:SESSion<n >:SMC:ECAL:CHARacteriza <mod> ,<char>

(Read-Write) Specifies the ECal module and characterization to be used for the SMC calibration.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<n> Session number of the calibration.

[Learn about Cal sessions.](#)

<mod> **1** - Electronic Calibration Module

<char> Specifies which characterization within the ECal module from which to read the confidence data.

0 Factory characterization (data that was stored in the ECal module by Agilent). Default if not specified.

1 User characterization #1

2 User characterization #2

3 User characterization #3

4 User characterization #4

5 User characterization #5

Examples

```
SENS:CORR:COLL:SESS:SMC:ECAL:CHAR 1,2
```

Query Syntax SENS:CORR:COLL:SESS:SMC:ECAL:CHAR?

Return Type Numeric

Default 1,0

SENSe<ch>:CORRection:COLLect:SESSion<n> :SMC:FSIMulator:NETWork<p>:FILename <string>

(Read-Write) Specifies the S2P filename to embed or de-embed on the input or output of your mixer measurement. [Learn more.](#)

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<n> Session number of the calibration.

[Learn about Cal sessions.](#)

<p> Apply network to input or output of mixer. Choose from:

1 - Input of mixer

2 - Output of mixer

<string> Filename of the S2P used for embedding or de-embedding. Use the full path name, file name, and .S2P suffix, enclosed in quotes.

Examples

```
SENS:CORR:COLL:SESS:SMC:FSIM:NETW1:FIL "c:\users\public\network analyzer\documents\WaveguideAdapt.S2P"
```

Query Syntax SENS<ch>:CORRection:COLLect:SESSion<n> :SMC:FSIMulator:NETWork<x>:FILename?

Return Type String

Default Not Applicable

SENSe<ch>:CORRection:COLLect:SESSion<n>:SMC:FSIMulator:NETWork<p>:MODE <char>

(Read-Write) Allows you to embed (add) or de-embed (remove) circuit network effects on the input and output of your mixer measurement. [Learn more.](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Session number of the calibration.

[Learn about Cal sessions.](#)

- <p> Apply network to input or output of mixer. Choose from:
 - 1 - Input of mixer
 - 2 - Output of mixer
- <char> Choose from:

NONE - Do nothing with effects of S2P file.

EMBed - Add effects of S2P file from the measurement results.

DEEMbed - Remove effects of S2P file from the measurement results.

Examples

```
SENS:CORR:COLL:SESS:SMC:FSIM:NETW1:MODE EMB
```

Query Syntax SENS<ch>:CORRection:COLLect:SESSion<n>
:SMC:FSIMulator:NETWork<x>:MODE?

Return Type Character

Default NONE

SENSe<ch>:CORRection:COLLect:SESSion<n>:SMC:IMPort <calName>, <dataset>

(Write-only) Imports existing Source Power Cal data into the SMC calibration.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Session number of the calibration.

[Learn about Cal sessions.](#)

- <calName> (String) Name of existing Cal Set from which power meter data is imported.
- <dataset> (String) Name of the data set. Use POWER_STEP

Examples

```
SENS2:CORR:COLL:SESS1:SMC:IMP "MyPowerCal", "POWER_STEP"
```

Query Syntax Not Applicable

Default NONE

SENSe<ch>:CORRection:COLLEct:SESSion<n>:SMC:PHASe:DELay <num>

(Read-Write) Set and return the known delay through the calibration mixer. [Learn more.](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Session number of the calibration.

[Learn about Cal sessions.](#)

- <char> Known delay through the calibration mixer in seconds.

Examples

```
SENS:CORR:COLL:SESS:SMC:PHAS:DEL 12e-9
```

```
sense2:correction:collect:session2:smc:phase:delay 12e-10
```

Query Syntax SENSe<ch>:CORRection:COLLEct:SESSion<n>:SMC:PHASe:DELay?

Return Type Numeric

Default 0 seconds

SENSe<ch>:CORRection:COLLEct:SESSion<n>:SMC:PHASe:METHod <char>

(Read-Write) Set and return the method of setting the delay through the calibration mixer. [Learn more.](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
<n> Session number of the calibration.

[Learn about Cal sessions.](#)

<char> **Choose from:**

FIXed - use a known delay value set with
[SENS:CORR:COLL:SESS:SMC:PHAS:DEL](#)

MIXer - use the S2P file set with
[SENS:CORR:COLL:SESS:SMC:PHAS:MIX](#)

Examples

```
SENS:CORR:COLL:SESS:SMC:PHAS:METH FIX
```

```
sense2:correction:collect:session2:smc:phase:method mixer
```

Query Syntax SENSE<ch>:CORRection:COLLect:SESSion<n>:SMC:PHASe:METhod?

Return Type Character

Default FIXed

SENSE<ch>:CORRection:COLLect:SESSion<n>:SMC:PHASe:MIXer <string>

(Write-only) Set the filename of the S2P file used to characterize the calibration mixer. [Learn more.](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
<n> Session number of the calibration.

[Learn about Cal sessions.](#)

<string> Calibration mixer filename. Use the following rules to specify path names:

- The default folder is "c:\users\public\network analyzer\documents"
- You can change the active directory using [MMEMory:CDIRectory](#).
- Specify only the file name if using the active directory.
- You can also use an absolute path name to specify the folder and file.

Examples

```
SENS:CORR:COLL:SESS:SMC:PHAS:MIX "MyCalMixer.s2p"
sense2:correction:collect:session2:smc:phase:mixer
"MyCalMixer.s2p"
```

Query Syntax Not Applicable**Default** Not Applicable**SENSe<ch>:CORRection:COLLect:SESSion<n>:SMC:PWRCal:SEParate <bool>**

(Read-Write) Specifies whether to use a Thru standard or to use two power sensor connections during the power cal of an SMC calibration.

This command must be sent BEFORE the [INITiate](#) command and all the other calibration commands.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<n> Session number of the calibration.

[Learn about Cal sessions.](#)

<bool> **OFF or 0** - Perform Cal with Thru standard.

ON or 1 - Do NOT use a Thru, but instead perform separate power cals on Input and Output reference planes.

Examples

```
SENS:CORR:COLL:SESS:INIT "SMC"
SENS:CORR:COLL:SESS:SMC:PWRC:SEP 1
```

Query Syntax SENS:CORR:COLL:SESS:SMC:PWRCal:SEParate?**Return Type** Boolean**Default** 0 or OFF**SENSe<ch>:CORRection:COLLect:SESSion<n>:SMC:PWRCal:SRCPort <string> **Obsolete****

(Read-Write) Specifies which port to calibrate.

Note: Beginning with Rev 6.0, this command is no longer necessary. Because of improved calibration techniques, **Both** is always selected although a power meter measurement is performed only on port 1.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Session number of the calibration.

[Learn about Cal sessions.](#)

- <char> **'1'** Source port 1 (SMC forward direction)
- '2'** Source port 2 (SMC reverse direction)
- 'BOTH'** (both forward and reverse directions)

Examples

```
SENS:CORR:COLL:SESS:SMC:PWRC:SRCP 'both'  
SENSe2:CORRection:COLLection:SESSion6:SMC:PWRCal:SRCPort '2'
```

Query Syntax SENS:CORR:COLL:SESS:SMC:PWRC:SRCP?

Return Type String

Default 1

**SENSe<ch>:CORRection:COLLection:SESSion<n>:SMC:TWOPort:ECAL:ORientation[:STATe]
<bool>**

(Read-Write) Sets ECAL Auto-Orientation ON or OFF. If setting auto-orientation OFF, you must manually specify the orientation of the ECAL module with

[SENS:CORR:COLL:SESS:SMC:TWOP:ECAL:PORTmap.](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Session number of the calibration.

[Learn about Cal sessions.](#)

- <bool> **OFF or 0** = Orientation OFF
- ON or 1** = Orientation ON

Examples

```
SENS:CORR:COLL:SESS:SMC:TWOP:ECAL:ORI 1
```

Query Syntax SENS:CORR:COLL:SESS:SMC:TWOP:ECAL:ORI?

Return Type Boolean

SENSe<ch>:CORRection:COLLEct:SESSion<n>:SMC:TWOPort:ECAL:PORTmap <mod>, <string>

(Read-Write) Specifies the manual orientation (which ports of the module are connected to which ports of the PNA) when [auto-orientation](#) is OFF.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Session number of the calibration.

[Learn about Cal sessions.](#)

<mod> **1** - Electronic Calibration Module

<string> Format in the following manner:

Aw,Bx,Cy,Dz

where

- A, B, C, and D are literal ports on the ECAL module
- w,x,y, and z are substituted for PNA port numbers to which the ECAL module port is connected.

Ports of the module which are not used are omitted from the string.

For example, on a 4-port ECal module with

- port A connected to PNA port 2
- port B connected to PNA port 3
- port C not connected
- port D connected to PNA port 1

the string would be: A2,B3,D1

If either the receive port or source port (or load port for 2-port cal) of the

CALC:PAR:SElected measurement is not in this string and orientation is OFF, an attempt to perform an ECal calibration will fail.

Examples

```
SENS:CORR:COLL:SESS:SMC:TWOP:ECAL:PORTmap 1, 'A1,B2'
```

Query Syntax

```
SENS:CORR:COLL:SESS:SMC:TWOP:ECAL:PORTmap?
```

Return Type

String

Default

"A1,B2"

SENSe<ch>:CORRection:COLLEct:SESSion<n >:SMC:TWOPort:MEthod <string>

(Read-Write) Specifies the guided ECal method for performing the thru portion of the calibration.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<n> Session number of the calibration.

[Learn about Cal sessions.](#)

<string> ECAL Method: Choose from:

'DEFAULT' - Default

'ADAP' - Adapter removal

'FLUSH' - Flush Through

'UNKN' - Unknown Thru

Examples

```
SENS:CORR:COLL:SESS:SMC:TWOPort:MEth 'default'
```

Query Syntax

```
SENS:CORR:COLL:SESS:SMC:TWOPort:MEth?
```

Return Type

String

Default

DEFAULT

SENSe<ch>:CORRection:COLLEct:SESSion<n >:SMC:TWOPort:OMITisolat <bool>

(Read-Write) Select to omit or perform the isolation portion of the ECAL.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Session number of the calibration.

[Learn about Cal sessions.](#)

- <bool> **ON or 1** - Omit isolation
OFF or 0 - Perform isolation

Examples

```
SENS:CORR:COLL:SESS:SMC:TWOPort:OMIT 1
```

Query Syntax SENS:CORR:COLL:SMC:TWOPort:OMIT?

Return Type Boolean

Default 1

SENSe<ch>:CORRection:COLLect:SESSion<n >:SMC:TWOPort:OPTion <string>

(Read-Write) Sets the SMC calibration to ECAL or MECHanical

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Session number of the calibration.

[Learn about Cal sessions.](#)

- <char> Choose from:

'ECAL' Electronic Calibration Module -

Note: This selection assumes there is only one ECal module on the USB and so it selects the first enumerated module on the bus, and the factory characterization on that ECal module, to be used for the cal.

'MECH' Mechanical Calibration Kit

Examples

```
SENS:CORR:COLL:SESS:SMC:TWOPort:OPTion 'ECAL'
```

Query Syntax SENS:CORR:COLL:SESS:SMC:TWOPort:OPTion?

Return Type String

Default ECAL

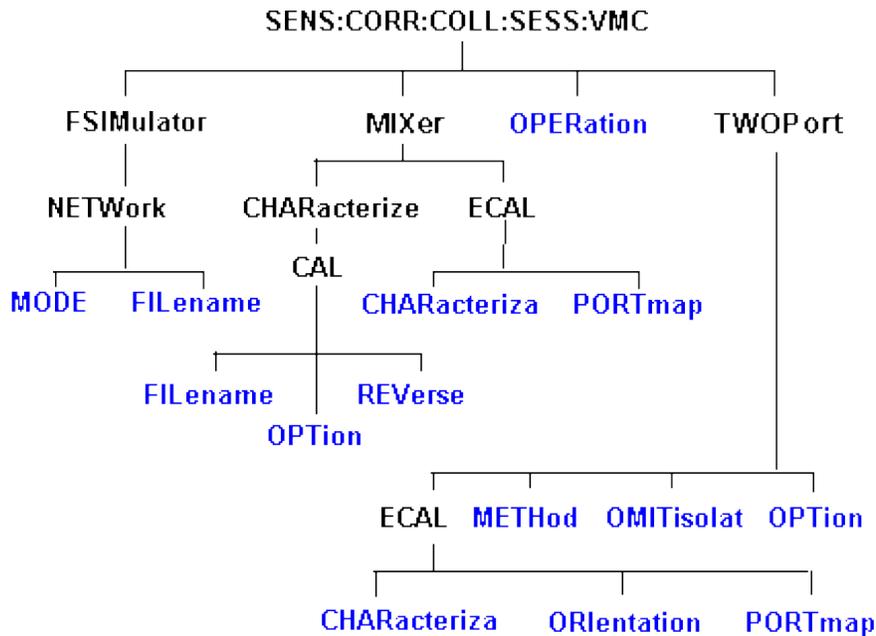
Last Modified:

9-Aug-2013	Added ORI back to the tree
9-Feb-2011	Added note to SMC:TWOPort:OPTion 'ECAL'
1-Dec-2010	Superseded (A.09.33)
22-Dec-2009	Added Import
24-Feb-2009	Replaced True/False
5-Aug-2008	Edited sessions description

Sense:Correction:Collect:Session:VMC Commands - Superseded

Note: Beginning with PNA release A..09.33, these commands (commonly known as "Session" commands) were replaced with [Sens:Corr:Coll:Guid:VMC](#) commands .

Performs a vector (VMC) calibration on a frequency converting device.



Click on a [blue](#) keyword to view the command details.

See Also

- [Example Programs](#)
- [Learn about VMC Calibration](#)
- [Synchronizing the PNA and Controller](#)

**SENSe<ch>:CORRection:COLLect:SESSion<n> :VMC:FSIMulator:NETWork<x>:MODE
<char>**

(Read-Write) Allows you to embed (add) or de-embed (remove) circuit network effects on the input and output of your mixer measurement. [Learn more.](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Session number of the calibration.

[Learn about Cal sessions.](#)

- <x> Apply network to input or output of mixer. Choose from:

1 - Input of mixer

2 - Output of mixer

- <char> Choose from:

NONE - Do nothing with effects of S2P file.

EMBed - Add effects of S2P file from the measurement results.

DEEMbed - Remove effects of S2P file from the measurement results.

Examples

```
SENS:CORR:COLL:SESS:VMC:FSIM:NETW1:MODE EMB
```

Query Syntax SENS<ch>:CORRection:COLLect:SESSion<n>
:VMC:FSIMulator:NETWork<x>:MODE?

Return Type Character

Default NONE

**SENSe<ch>:CORRection:COLLect:SESSion<n> :VMC:FSIMulator:NETWork<x>:FILEname
<string>**

(Read-Write) Specifies the S2P filename to embed or de-embed on the input or output of your mixer measurement. [Learn more.](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Session number of the calibration.

[Learn about Cal sessions.](#)

- <x> Apply network to input or output of mixer. Choose from:
 - 1 - Input of mixer
 - 2 - Output of mixer
- <string> Filename of the S2P used for embedding or de-embedding. Use the full path name, file name, and .S2P suffix, enclosed in quotes.

Examples

```
SENS:CORR:COLL:SESS:VMC:FSIM:NETW1:FIL "c:\users\public\network analyzer\documents\WaveguideAdapt.S2P"
```

Query Syntax SENS<ch>:CORRection:COLLect:SESSion<n>:VMC:FSIMulator:NETWork<x>:FILename?

Return Type String

Default Not Applicable

SENSe<ch>:CORRection:COLLect:SESSion<n> :VMC:MIXer:CHARacterize:CAL:FILename <string>

(Read-Write) Specifies the .S2P filename used for mixer characterization. Use the [VMC:MIXer:CHARacterize:CAL: OPTion](#) command to load the file for mixer characterization. Once loaded, use this command to query the current filename or set a new filename.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Session number of the calibration.

[Learn about Cal sessions.](#)

- <string> Filename of the S2P used for mixer characterization. Use the full path name, file name, and .S2P suffix, enclosed in quotes.

Examples

```
SENS2:CORR:COLL:SESS4:VMC:MIXer:CHAR:CAL:FIL "c:\users\public\network analyzer\documents\MyMixer.S2P"
```

Query Syntax SENS:CORR:COLL:SESS:VMC:MIX:CHAR:CAL:FIL?

Return Type String

Default c:\users\public\network analyzer\documents/default.s2p

SENSe<ch>:CORRection:COLLect:SESSion<n> :VMC:MIXer:CHARacterize:CAL: OPTion <char>

(Read-Write) Sets the mixer characterization method to ECal, Mechanical, or read from a file.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Session number of the calibration.

[Learn about Cal sessions.](#)

<char> **ECAL** - Electronic Calibration Module

MECH - Mechanical Calibration Kit

FILE, <filename> - Retrieve a mixer characterization file. Also specify the filename of the S2P used for mixer characterization. Use the full path name, file name, and .S2P suffix. Use the [VMC:CHARacterize:CAL:FILEname](#) command to query the filename..

Examples

```
SENS:CORR:COLL:SESS:VMC:MIX:CHAR:CAL:OPT ECAL
```

```
'or
```

```
SENS:CORR:COLL:SESS6:VMC:MIX:CHAR:CAL:OPT
```

```
FILE,'c:\users\public\network analyzer\documents\Mixer001.s2p'
```

```
file = "SENS:CORR:COLL:SESS6:VMC:MIXer:CHAR:CAL:FIL?" 'Read back  
the filename
```

Query Syntax SENS:CORR:COLL:SESS:VMC:MIX:CHAR:CAL:OPT?

Return Type String

Default MECH

SENSe<ch>:CORRection:COLLect:SESSion<n> :VMC:MIXer:CHARacterize:CAL: REVerse <bool>

(Read-Write) Specifies the direction in which to characterize the calibration mixer. [Learn more about the calibration mixer.](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Session number of the calibration.

[Learn about Cal sessions.](#)

- <bool> **OFF (0)** - Characterize the calibration mixer in the SAME direction as that specified in the mixer setup.
ON (1) - Characterize the calibration mixer in the REVERSE direction as that specified in the mixer setup.

Examples

```
SENS:CORR:COLL:SESS:VMC:MIX:CHAR:CAL:REV 1
```

Query Syntax SENS:CORR:COLL:SESS:VMC:MIX:CHAR:CAL:REV?

Return Type Boolean

Default OFF

**SENSe<ch>:CORRection:COLLEct:SESSion<n > :VMC:MIXer:ECAL:CHARacteriza <mod>
,<char>**

(Read-Write) Specifies the ECal module and characterization to be used for the mixer characterization portion of the calibration.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Session number of the calibration.

[Learn about Cal sessions.](#)

- <mod> **1** - Electronic Calibration Module
- <char> Characterization number in the specified ECAL module. Choose from:
 - 0** Factory characterization (data that was stored in the ECal module by Agilent). Default if not specified.
 - 1** User characterization #1
 - 2** User characterization #2
 - 3** User characterization #3

4 User characterization #4

5 User characterization #5

Examples `SENS:CORR:COLL:SESS:VMC:MIX:ECAL:CHAR 1,0`

Query Syntax `SENS:CORR:COLL:SESS:VMC:MIX:ECAL:CHAR?`

Return Type Numeric

Default 1,0

SENSe<ch>:CORRection:COLLect:SESSion<n> :VMC:MIXer:ECAL:PORTmap <mod>, <string>

(Read-Write) Sets the port mapping for the mixer characterization with ECAL. This command is required if [SENS:CORR:COLL:SESS:VMC:MIX:CHAR:CAL:OPT ECAL](#) is specified.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<n> Session number of the calibration.

[Learn about Cal sessions.](#)

<mod> **1** - Electronic Calibration Module

<string> Choose from:

"A1" - ECAL module port A is connected to PNA port 1

"B1" - ECAL module port B is connected to PNA port 1

Examples `SENS:CORR:COLL:SESS:VMC:MIX:ECAL:PORT?`
`SENS2:CORR:COLL:SESS:VMC:MIXer:ECAL:PORTmap 1,"A1"`

Query Syntax `SENS:CORR:COLL:SESS:VMC:MIX:ECAL:PORTmap?`

Return Type String

Default "A1"

SENSe<ch>:CORRection:COLLect:SESSion<n>:VMC:OPERation <string>

(Read-Write) Perform either full VMC calibration or mixer characterization only.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Session number of the calibration.

[Learn about Cal sessions.](#)

- <char> **'CAL'** - full calibration and mixer characterization
- 'CHAR'** - mixer characterization only (no reference mixer required) - Saves an .S2P file with the filename specified in [SENS<ch>:CORR:COLL:SESSion<n>:VMC:CHARacterize:CAL:FILEname<filename>](#). If none is specified, a filename is automatically generated and can be queried using the filename command.

Examples

```
SENS:CORR:COLL:SESS:VMC:OPER 'CAL'
```

Query Syntax SENS:CORR:COLL:SESS:VMC:OPER?

Return Type String

Default CAL

SENSe<ch>:CORRection:COLLEct:SESSion<n> :VMC:TWOPort:ECAL:CHARacteriza <mod> ,<char>

(Read-Write) Specifies the ECal module and characterization to be used for the VMC calibration.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Session number of the calibration.

[Learn about Cal sessions.](#)

- <mod> **1** - Electronic Calibration Module
- <char> Characterization number in the specified ECAL module. Choose from:
 - 0** Factory characterization (data that was stored in the ECal module by Agilent). Default if not specified.
 - 1** User characterization #1
 - 2** User characterization #2
 - 3** User characterization #3

4 User characterization #4

5 User characterization #5

Examples SENS:CORR:COLL:SESS:VMC:TWOP:ECAL:CHAR 1,1

Query Syntax SENS:CORR:COLL:SESS:VMC:TWOP:ECAL:CHAR?

Return Type Integer

Default 1,0

**SENSe<ch>:CORRection:COLLEct:SESSion<n>:VMC:TWOPort:ECAL:ORientation:STATe
<bool>**

(Read-Write) Sets ECAL orientation for the VMC ECal.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<n> Session number of the calibration.

[Learn about Cal sessions.](#)

<bool> **ON or 1** - Perform orientation

OFF or 0 - Do NOT perform orientation

Examples SENS:CORR:COLL:SESS:VMC:TWOP:ECAL:ORI:STAT 1

Query Syntax SENS:CORR:COLL:SESSion:VMC:TWOPort:ECAL:ORientation:STATe?

Return Type Integer

Default ON

**SENSe<ch>:CORRection:COLLEct:SESSion<n> :VMC:TWOPort:ECAL:PORTmap <mod>,
<string>**

(Read-Write) Specifies the manual orientation (which ports of the module are connected to which ports of the PNA) when [orientation](#) is turned off.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Session number of the calibration.

[Learn about Cal sessions.](#)

- <mod> **1** - Electronic Calibration Module
- <string> Port Map, formatted in the following manner:

Aw,Bx,Cy,Dz

where:

- A, B, C, and D are literal ports on the ECAL module.
- w,x,y, z are substituted for PNA port numbers to which the ECAL module port is connected.
- Ports of the module which are not used are omitted from the string.

For example, on a 4-port ECal module with:

- port A connected to PNA port 2
- port B connected to PNA port 3
- port C not connected
- port D connected to PNA port 1

the string would be: A2,B3,D1

If either the receive port or source port (or load port for 2-port cal) of the measurement is not in this string and orientation is OFF, an attempt to perform an ECal will fail.

Examples

```
SENS:CORR:COLL:SESS:VMC:TWOP:ECAL:PORTmap 1,"A2,B1"
```

Query Syntax SENS:CORR:COLL:SESS:VMC:TWOP:ECAL:PORTmap?

Return Type string

Default "A1,B2"

SENSe<ch>:CORRection:COLLEct:SESSion<n> :VMC:TWOPort:METhod <string>

(Read-Write) Specifies the guided ECal method for performing the thru portion of the calibration.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Session number of the calibration.

[Learn about Cal sessions.](#)

- <char> 'DEFAULT' - Default
- 'ADAP' - Adapter removal
- 'FLUSH' - Flush Through
- 'UNKN' - Unknown Thru

Examples

```
SENS:CORR:COLL:SESS:VMC:TWOP:METH 'ADAP'  
SENSe2:CORR:COLL:SESSion6:VMC:TWOPort:METhod 'FLUSH'
```

Query Syntax SENS:CORR:COLL:SESS:VMC:TWOP:METH?

Return Type String

Default DEFAULT

SENSe<ch>:CORRection:COLLEct:SESSion<n> :VMC:TWOPort:OMITisolat <bool>

(Read-Write) Select to omit or perform the isolation portion of the ECAL.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Session number of the calibration.

[Learn about Cal sessions.](#)

- <bool> ON or 1 - omit isolation
- OFF or 0 - perform isolation

Examples

```
SENS:CORR:COLL:SESS:VMC:TWOP:OMIT 1
```

Query Syntax SENS:CORR:COLL:SESS:VMC:TWOP:OMIT?

Return Type Boolean

Default ON

SENSe<ch>:CORRection:COLLEct:SESSion<n> :VMC:TWOPort:OPTion <string>

(Read-Write) Sets the 2-port calibration option to ECAL or MECHanical

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Session number of the calibration.

[Learn about Cal sessions.](#)

<char> **Choose from:**

'ECAL' Electronic Calibration Module

'MECH' Mechanical Calibration Kit

Examples

```
SENS:CORR:COLL:SESS:VMC:TWOP:OPT 'MECH'  
SENSe2:CORR:COLL:SESSion6:VMC:TWOPort:OPTion 'ECAL'
```

Query Syntax SENS:CORR:COLL:SESS:VMC:TWOP:OPT?

Return Type String

Default "MECH"

Last Modified:

1-Dec-2010	Superseded (A.09.33)
24-Feb-2009	Replaced True/False
5-Aug-2008	Edited session description
29-Aug-2007	Edited Char:Cal:Opt File argument

Sense:Couple Commands

SENSe:COUPle

|
PARAmeter

|
[STATe]

Click on a [blue](#) keyword to view the command details.

See Also

- [Example Programs](#)
 - [Synchronizing the PNA and Controller](#)
 - [SCPI Command Tree](#)
-

SENSe<num>:COUPle <ALL | NONE>

(Read-Write) Sets the sweep mode as Chopped or Alternate.

[Learn about Alternate Sweep](#)

Parameters

<num> Any existing channel number; if unspecified, value is set to 1.

<ALL | NONE> **ALL** - Sweep mode set to Chopped - reflection and transmission measured on the same sweep.

NONE - Sweep mode set to Alternate - reflection and transmission measured on separate sweeps. Improves Mixer bounce and Isolation measurements.
Increases sweep time

Examples

```
SENS:COUP ALL  
sense2:couple none
```

Query Syntax SENSe<num>:COUPle?

Return Type Character

Default ALL

SENSe<num>:COUPle:PARAmeter[:STATe] <bool>

(Read-Write) Turns ON and OFF Time Domain Trace Coupling. All of the measurements in the specified channel are coupled.

- To select Transform parameters to couple, use [CALC:TRAN:COUP:PAR](#)
- To select Gating parameters to couple, use [CALC:FILT:COUP:PAR](#)

Learn more about [Time Domain Trace Coupling](#).

Parameters

<num> Any existing channel number; if unspecified, value is set to 1.

<bool> **ON (or 1)** - Turns ON Time Domain Trace Coupling.

OFF (or 0) - Turns OFF Time Domain Trace Coupling.

Examples

```
SENS:COUP:PAR 0  
sense2:couple:parameter:state on
```

Query Syntax SENSE<num>:COUPLE:PARAMETER[:STATE]?

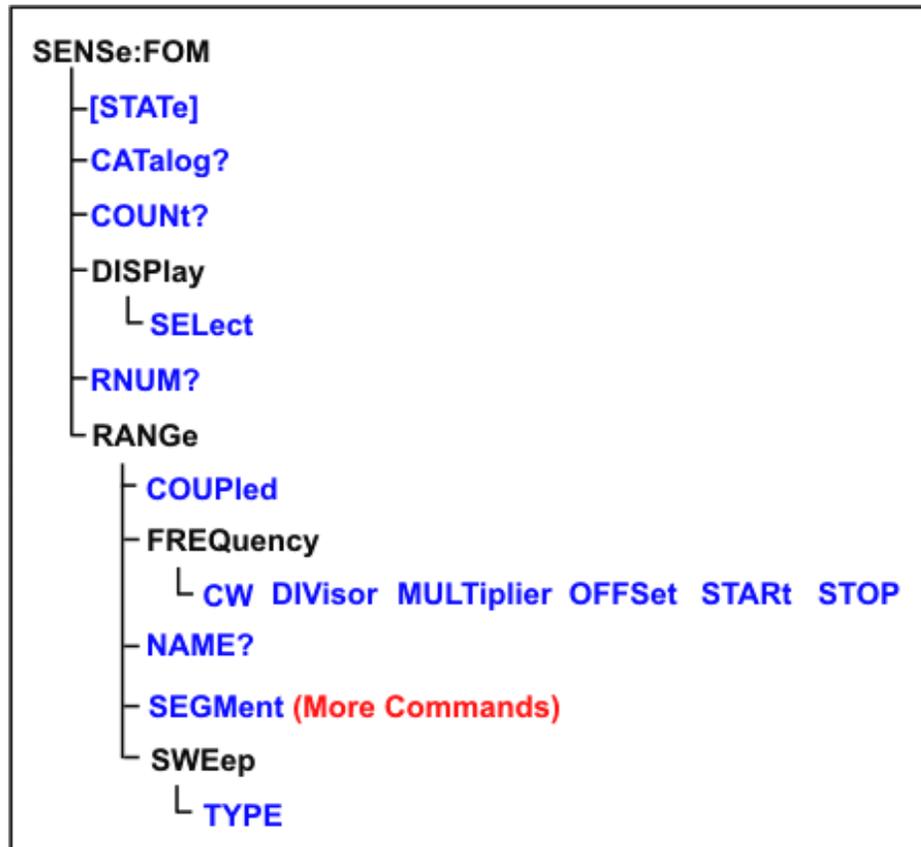
Return Type Boolean

Default ON (or 1)

Sense:FOM (Frequency Offset) Commands

Controls the frequency offset settings which cause stimulus and response frequencies to be different.

Note: These commands replace the [previous FOM commands](#). Although the old commands will continue to work, they can NOT be mixed with these new commands.



Click on a [blue](#) keyword to view the command details.

See Also

- [FOM Example Program](#)
- [Learn about Frequency Offset](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

SENSE<num>:FOM[:STATe] <bool>

(Read-Write) Turns Frequency Offset ON and OFF. Frequency offset settings are not enabled until this setting is ON.

Send this command (FOM ON) AFTER sending other FOM settings to avoid 'out-of-range' errors.

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1.

<bool> ON (or 1) - turns FOM ON.

OFF (or 0) - turns FOM OFF.

Examples

```
SENS:FOM 1
sense2:fom:state on
```

Query Syntax SENSE<cnum>:FOM:STATE?

Return Type Boolean

Default OFF

SENSe<cnum>:FOM:CATalog?

(Read-only) Returns a comma-separated list of available range names in the PNA.

Use [SENS:FOM:CAT?](#) to see a list of available range names.

Use [SENS:FOM:COUNT?](#) to see a list of available range numbers.

Use [SENS:FOM:RNUM?](#) to see the range number for a specific name.

Use [SENS:FOM:RANG:NAME?](#) to see the range name for a specific number.

External devices can appear in the list of range names. [Learn more.](#)

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1.

Examples

```
SENS:FOM:CAT?
'returns
"Primary, Source, Receivers"
```

Return Type String

Default Not Applicable

SENSe<cnum>:FOM:COUNT?

(Read-only) Returns the number of valid range numbers in the PNA.

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1.

Examples

```
SENS:FOM:COUN?  
sense2:fom:count?
```

Query Syntax SENSE<cnum>:FOM:COUNT?

Return Type Numeric

Default Not Applicable

SENSE<cnum>:FOM:DISPlay:SElect <string>

(Read-Write) Select the range to be displayed on the PNA x-axis. All traces in the channel have this same x-axis scaling.

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1.

<string> Range name. Case insensitive. Use [SENSE:FOM:CAT?](#) to see a list of available frequency range names.

Examples

```
SENS:FOM:DISPlay:SElect "source2"  
sense2:fom:display:select "source"
```

Query Syntax SENSE<cnum>:FOM:DISPlay:SElect?

Return Type String

Default Receivers

SENSE<cnum>:FOM:RNUM? <string>

(Read-only) Returns the number of a specified range name.

The FOM range items are typically numbered as follows:

1. Primary
2. Source
3. Receivers
4. Source2 (if present)

Use [SENS:FOM:CAT?](#) to see a list of available range names.

Use [SENS:FOM:COUNT?](#) to see a list of available range numbers.

Use [SENS:FOM:RANG:NAME?](#) to see the range name for a specific number.

External devices can appear in the list of range names. [Learn more.](#)

Parameters

- <cnum> Any existing channel number; if unspecified, value is set to 1.
- <string> Range name for which a number is being queried. Case insensitive.

Examples

```
SENS:FOM:RNUM? "receivers"  
sense2:fom:rnum? "Source2"
```

Return Type Numeric

Default Not Applicable

SENSe<cnum>:FOM:RANGe<n>:COUPlEd <bool>

(Read-Write) Sets and returns the state of coupling (ON or OFF) of the specified range to the primary range.

Parameters

- <cnum> Any existing channel number; if unspecified, value is set to 1.
- <n> Range number to couple to primary range. An error is returned when attempting to couple to the Primary range (1).

Use [SENS:FOM:CAT?](#) to see a list of available range names.

Use [SENS:FOM:COUNT?](#) to see a list of available range numbers.

Use [SENS:FOM:RNUM?](#) to see the range number for a specific name.

Use [SENS:FOM:RANG:NAME?](#) to see the range name for a specific number.

<bool> **ON** (or 1) - Couple range to primary range.

OFF (or 0) - Do NOT couple to primary range.

Examples

```
SENS:FOM:RANG:COUP 1
sense2:fom:range2:coupled 0
```

Query Syntax SENSE<cnum>:FOM:RANGe<n>:COUPled?

Return Type Boolean

Default ON (or 1) Coupled

SENSe<cnum>:FOM:RANGe<n>:FREQUency:CW <num>

(Read-Write) Sets and returns the CW frequency.

This setting is valid for the primary range, or if the specified range is already [uncoupled](#) from the primary range and if the [sweep type](#) is CW.

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1.

<n> Range number. If unspecified, value is set to 1.

Use [SENS:FOM:CAT?](#) to see a list of available range names.

Use [SENS:FOM:COUNT?](#) to see a list of available range numbers.

Use [SENS:FOM:RNUM?](#) to see the range number for a specific name.

Use [SENS:FOM:RANG:NAME?](#) to see the range name for a specific number.

<num> CW frequency value in Hz. Choose any frequency within the range of the PNA.

Examples

```
SENS:FOM:RANG:FREQ:CW 1e9
sense2:fom:range2:frequency:cw 10000000
```

Query Syntax SENSE<cnum>:FOM:RANGe:<n>:FREQUency:CW?

Return Type Numeric

Default Center frequency of the PNA.

SENSe<cnum>:FOM:RANGe<n>:FREQUency:DIVisor <num>

(Read-Write) Sets and returns the divisor value.

This setting is valid only if the specified range is [coupled](#) to the primary range.

Parameters

<cnun> Any existing channel number; if unspecified, value is set to 1.

<n> Range number. If unspecified, value is set to 1.

Use [SENS:FOM:CAT?](#) to see a list of available range names.

Use [SENS:FOM:COUNt?](#) to see a list of available range numbers.

Use [SENS:FOM:RNUM?](#) to see the range number for a specific name.

Use [SENS:FOM:RANG:NAME?](#) to see the range name for a specific number.

<num> Divisor value (unitless).

Examples

```
SENS:FOM:RANG:FREQ:DIV 3
sense2:fom:range2:frequency:divisor 0
```

Query Syntax SENSE<cnun>:FOM:RANGe<n>:FREQUency:DIVisor?

Return Type Numeric

Default 1

SENSE<cnun>:FOM:RANGe<n>:FREQUency:MULTIplier <num>

(Read-Write) Sets and returns the multiplier value to be used when coupling this range to the primary range.

This setting is valid only if the specified range is [coupled](#) to the primary range.

Parameters

<cnun> Any existing channel number; if unspecified, value is set to 1.

<n> Range number. If unspecified, value is set to 1.

Use [SENS:FOM:CAT?](#) to see a list of available range names.

Use [SENS:FOM:COUNt?](#) to see a list of available range numbers.

Use [SENS:FOM:RNUM?](#) to see the range number for a specific name.

Use [SENS:FOM:RANG:NAME?](#) to see the range name for a specific number.

<num> Multiplier value. (Unitless)

Examples

```
SENS:FOM:RANG:FREQ:MULT 1
sense2:fom:range2:frequency:multiplier 2
```

Query Syntax SENSE<cnum>:FOM:RANGe<n>:FREQuency:MULTplier?

Return Type Numeric

Default 1

SENSe<cnum>:FOM:RANGe<n>:FREQuency:OFFSet <num>

(Read-Write) Sets and returns the offset value to be used when coupling this range to the primary range. [Learn more about offset value.](#)

This setting is valid only if the specified range is [coupled](#) to the primary range.

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1.

<n> Range number. If unspecified, value is set to 1.

Use [SENS:FOM:CAT?](#) to see a list of available range names.

Use [SENS:FOM:COUNT?](#) to see a list of available range numbers.

Use [SENS:FOM:RNUM?](#) to see the range number for a specific name.

Use [SENS:FOM:RANG:NAME?](#) to see the range name for a specific number.

<num> Offset value. (Unitless)

Examples

```
SENS:FOM:RANG:FREQ:OFFS 1E9
sense2:fom:range2:frequency:offset 10000000
```

Query Syntax SENSE<cnum>:FOM:RANGe<n>:FREQuency:OFFSet?

Return Type Numeric

Default 0

SENSe<cnum>:FOM:RANGe<n>:FREQuency:STARt <num>

(Read-Write) Sets and returns the Start value of frequency range. Also specify [Stop frequency](#).

This setting is valid for the primary range, or if the specified range is already [uncoupled](#) from the primary range and if the [sweep type](#) is LOG or LINear.

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1.

<n> Range number. If unspecified, value is set to 1.

Use [SENS:FOM:CAT?](#) to see a list of available range names.

Use [SENS:FOM:COUNT?](#) to see a list of available range numbers.

Use [SENS:FOM:RNUM?](#) to see the range number for a specific name.

Use [SENS:FOM:RANG:NAME?](#) to see the range name for a specific number.

<num> Start value in Hz. Choose any frequency within the range of the PNA.

Examples

```
SENS:FOM:RANG:FREQ:STAR 1GHz  
sense2:fom:range2:frequency:start 100000000
```

Query Syntax SENSE<cnum>:FOM:RANGe<n>:FREQuency:STARt?

Return Type Numeric

Default Minimum frequency of the PNA.

SENSe<cnum>:FOM:RANGe<n>:FREQuency:STOP <num>

(Read-Write) Sets and returns the Stop value of frequency range. Also specify [Start frequency](#).

This setting is valid for the primary range, or if the specified range is already [uncoupled](#) from the primary range and if the [sweep type](#) is LOG or LINear.

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1.

<n> Range number. If unspecified, value is set to 1.

Use [SENS:FOM:CAT?](#) to see a list of available range names.

Use [SENS:FOM:COUNT?](#) to see a list of available range numbers.

Use [SENS:FOM:RNUM?](#) to see the range number for a specific name.

Use [SENS:FOM:RANG:NAME?](#) to see the range name for a specific number.

<num> Stop value in Hz. Choose any frequency within the range of the PNA.

Examples

```
SENS:FOM:RANG:FREQ:STOP 1e12
sense2:fom:range2:frequency:stop 10000000000
```

Query Syntax SENSE<cnum>:FOM:RANGe<n>:FREQuency:STOP?**Return Type** Numeric**Default** Maximum frequency of the PNA.**SENSe<cnum>:FOM:RANGe<n>:NAME?****(Read-only)** Returns the name of range<n>.

The FOM range items are typically named as follows:

1. Primary
2. Source
3. Receivers
4. Source2 (if present)

Use [SENS:FOM:CAT?](#) to see a list of available range names.Use [SENS:FOM:COUNT?](#) to see a list of available range numbers.Use [SENS:FOM:RNUM?](#) to see the range number for a specific name.**Parameters**

<cnum> Any existing channel number; if unspecified, value is set to 1.

<n> Range number. If unspecified, value is set to 1.

Examples

```
SENS:FOM:RANG:NAME?
sense2:fom:range2:name?
```

Return Type String**Default** Not Applicable**SENSe<cnum>:FOM:RANGe<n>:SWEep:TYPE <char>**

(Read-Write) Sets and returns the sweep type to be used with the specified range.

This setting is valid only if the specified range is already [uncoupled](#) from the primary range.

Learn about [Unsupported Sweep Type combinations](#).

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1.

<n> Range number. If unspecified, value is set to 1.

Use [SENS:FOM:CAT?](#) to see a list of available range names.

Use [SENS:FOM:COUNT?](#) to see a list of available range numbers.

Use [SENS:FOM:RNUM?](#) to see the range number for a specific name.

Use [SENS:FOM:RANG:NAME?](#) to see the range name for a specific number.

<char> Sweep type. Choose from:

CW - Also specify [CW frequency](#).

LINEar - Also specify frequency Start/Stop or Center/Span

LOG - Also specify frequency Start/Stop or Center/Span

PHASe - See all [Phase sweep](#) settings.

POWer - Also specify power Start/Stop or Center/Span

SEGMENT - Also specify [segment sweep](#) settings.

Examples

```
SENS:FOM:RANG:SWE:TYPE LOG  
sense2:fom:range2:sweep:type linear:
```

Query Syntax SENSE<cnum>:FOM:RANGe<n>:SWEep:TYPE?

Return Type Character

Default Linear

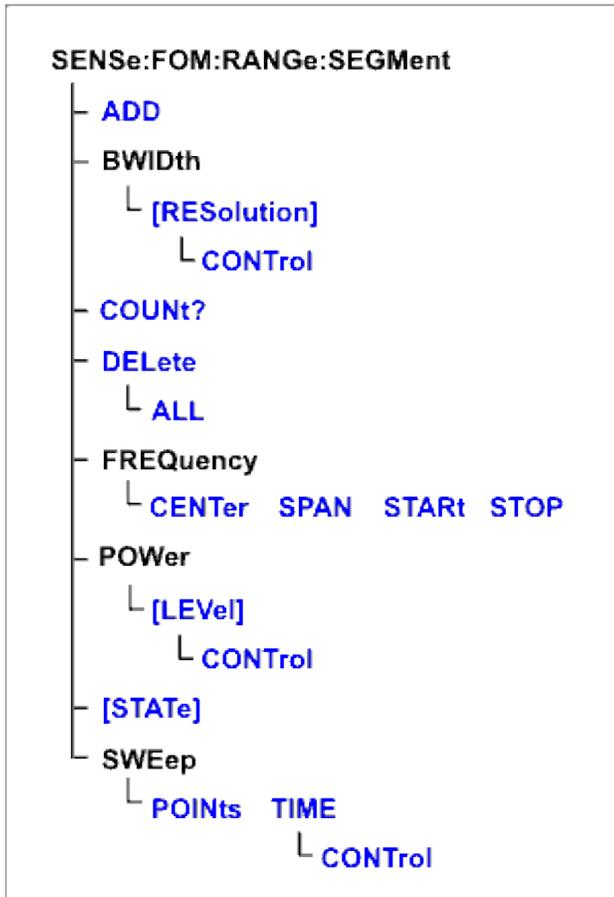
Last Modified:

7-Feb-2011	Added phase sweep
24-Sep-2008	Updated with range queries
5-Jul-2007	Fixed state link
8-Mar-2007	MQX New topic

Sense:FOM:Range:Segment Commands

Constructs a segment table for a specified [UNCOUPLED](#) FOM range.

Note: Do NOT use [Sens:Segment](#) commands for FOM segment sweep.



Click on a [blue](#) keyword to view the command details.

See Also

- [Other SENSE:FOM Commands](#)
- [Example Programs](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

SENSE<cnum>:FOM:RANGE<n>:SEGMENT<s>:ADD

(Write-only) Adds a segment.

Parameters

- <cnum> Any existing channel number; if unspecified, value is set to 1.
- <n> Range number. If unspecified, value is set to 1.
- <s> Segment number to add. If unspecified, value is set to 1. Segment numbers must be sequential. If a new number is added where one currently exists, the existing segment and those following are incremented by one.

Examples

Two Segments exist (1 and 2). The following command will add a new segment (1). The existing (1 and 2) will become (2 and 3) respectively.

```
sense2:fom:range2:segment:add
```

Query Syntax Not Applicable

Default Not Applicable

SENSe<cnum>:FOM:RANGe<n>:SEGMENT<s>:BWIDth[:RESolution] <num>

(Read-Write) Sets the IF Bandwidth for the specified segment. First set [SENS:FOM:RANGe:SEGM:BWIDth:CONTRol ON](#). All subsequent segments that are added assume the new IF Bandwidth value.

Valid either for Receiver range or for Primary range when coupled to Receiver.

Parameters

- <cnum> Any existing channel number; if unspecified, value is set to 1.
- <n> Range number. If unspecified, value is set to 1.
- <s> Segment number for which to set independent IF Bandwidth.
- <num> IF Bandwidth in Hz. The list of valid IF Bandwidths is different depending on the PNA model. [See the list of valid IFBW values](#). If an invalid number is specified, the analyzer will round up to the closest valid number.

Examples

```
SENS:FOM:RANG:SEGM:BWIDth 100
```

```
sense2:fom:range2:segment4:bandwidth:resolution 1e3
```

Query Syntax SENSe<cnum>:FOM:RANGe<n>:SEGMENT<s>:BWIDth[:RESolution]?

Return Type Numeric

Default Varies with model.

SENSe<cnum>:FOM:RANGe<n>:SEGMENT:BWIDth[:RESolution]:CONTRol <bool>

(Read-Write) Specifies whether the IF Bandwidth resolution can be set independently for each segment. When set, each segment added after this will be set to ON automatically.

Valid either for Receiver range or for Primary range. Primary range value is ignored unless Receiver is coupled to Primary.

Parameters

- <num> Any existing channel number; if unspecified, value is set to 1.
- <n> Range number. If unspecified, value is set to 1.
- <bool> **ON** (or 1) - turns Bandwidth control ON. Bandwidth can be set for each segment
OFF (or 0) - turns Bandwidth control OFF. Use the channel IF bandwidth setting instead.

Examples

```
SENS:FOM:RANG:SEGM:BWIDth:CONT 0
```

```
sense2:fom:range2:segment:bwidth:resolution:control 1
```

Query Syntax SENSE<num>:FOM:RANGe<n>:SEGMent:BWIDth[:RESolution]:CONTrol?

Return Type Boolean

Default OFF

SENSE<num>:FOM:RANGe<n>:SEGMent:COUNT?

(Read-only) Returns the number of segments that exist for the specified range.

Parameters

- <num> Any existing channel number; if unspecified, value is set to 1.
- <n> Range number. If unspecified, value is set to 1.

Examples

```
SENS:FOM:RANG:SEGM:COUN?
```

```
sense2:fom:range2:segment:count?
```

Return Type Numeric

Default Not Applicable

SENSE<num>:FOM:RANGe<n>:SEGMent<s>:DELeTe

(Write-only) Deletes the specified sweep segment.

Parameters

- <cnum> Any existing channel number; if unspecified, value is set to 1.
- <n> Range number. If unspecified, value is set to 1.
- <s> Number of the segment to delete. If unspecified, value is set to 1.

Examples

```
SENS:FOM:RANG:SEGM3:DEL
sense2:fom:range2:segment4:delete
```

Query Syntax Not Applicable

Default Not Applicable

SENSe<cnum>:FOM:RANGe<n>:SEGMENT:DELeTe:ALL

(Write-only) Deletes all sweep segments in the specified range.

Parameters

- <cnum> Any existing channel number; if unspecified, value is set to 1.
- <n> Range number. If unspecified, value is set to 1.

Examples

```
SENS:FOM:RANG:SEGM:DEL:ALL
sense2:fom:range2:segment:delete:all
```

Query Syntax Not Applicable

Default Not Applicable

SENSe<cnum>:FOM:RANGe<n>:SEGMENT<s>:FREQUency:CENTer <num>

(Read-Write) Sets and returns the center frequency for the specified sweep segment. Also specify segment frequency span.

Parameters

- <cnum> Any existing channel number; if unspecified, value is set to 1.
- <n> Range number. If unspecified, value is set to 1.
- <s> Segment number to modify. Choose any existing segment number.
- <num> Center Frequency in Hz. Choose any number between the minimum and maximum frequency of the analyzer.

Examples

```
SENS:FOM:RANG:SEGM:FREQ:CENT 1GHz
sense2:fom:range2:segment4:frequency:center 1e9
```

Query Syntax SENSE<num>:FOM:RANGe<n>:SEGMENT<s>:FREQUency:CENTer?

Return Type Numeric

Default Stop Frequency of the previous segment. If first segment, start frequency of the analyzer.

SENSe<num>:FOM:RANGe<n>:SEGMENT<s>:FREQUency:SPAN <num>

(Read-Write) Sets and returns the span frequency for the specified sweep segment. Also specify segment center frequency.

Parameters

- <num> Any existing channel number; if unspecified, value is set to 1.
- <n> Range number. If unspecified, value is set to 1.
- <s> Segment number to modify. Choose any existing segment number.
- <num> Frequency span in Hz. Choose any number between the minimum and maximum frequency of the analyzer.

Examples

```
SENS:FOM:RANG:SEGM:FREQ:SPAN 1GHz
```

```
sense2:fom:range2:segment4:frequency:span 1e9
```

Query Syntax SENSE<num>:FOM:RANGe<n>:SEGMENT<s>:FREQUency:SPAN?

Return Type Numeric

Default If first segment, frequency span of the analyzer. Otherwise 0.

SENSe<num>:FOM:RANGe<n>:SEGMENT<s>:FREQUency:START <num>

(Read-Write) Sets and returns the start frequency for the specified sweep segment. Also specify segment stop frequency.

All other segment Start and Stop Frequency values that are larger than this frequency are changed to this frequency.

Parameters

- <num> Any existing channel number; if unspecified, value is set to 1.
- <n> Range number. If unspecified, value is set to 1.
- <s> Segment number to modify. Choose any existing segment number.
- <num> Start frequency in Hz. Choose any number between the minimum and maximum frequency of the analyzer.

Examples `SENS:FOM:RANG:SEGM:FREQ:STAR 1GHz`

```
sense2:fom:range2:segment4:frequency:start 1e9
```

Query Syntax `SENSe<cnum>:FOM:RANGe<n>:SEGMent<s>:FREQUency:STARt?`

Return Type Numeric

Default Stop Frequency of the previous segment. If first segment, start frequency of the analyzer.

SENSe<cnum>:FOM:RANGe<n>:SEGMent<s>:FREQUency:STOP <num>

(Read-Write) Sets and returns the stop frequency for the specified sweep segment. Also specify segment start frequency.

All other segment Start and Stop Frequency values that are larger than this frequency are changed to this frequency.

Parameters

- <cnum> Any existing channel number; if unspecified, value is set to 1.
- <n> Range number. If unspecified, value is set to 1.
- <s> Segment number to modify. Choose any existing segment number.
- <num> Stop frequency in Hz. Choose any number between the minimum and maximum frequency of the analyzer.

Examples `SENS:FOM:RANG:SEGM:FREQ:STOP 1GHz`

```
sense2:fom:range2:segment4:frequency:stop 1e9
```

Query Syntax `SENSe<cnum>:FOM:RANGe<n>:SEGMent<s>:FREQUency:STOP?`

Return Type Numeric

Default Stop Frequency of the previous segment. If first segment, start frequency of the analyzer.

SENSe<cnum>:FOM:RANGe<n>:SEGMent<s>:POWER<p>[:LEVel] <num>

(Read-Write) Sets the Port Power level for the specified sweep segment. First set SENS:FOM:RANG:SEGM:POW:CONTRol ON.

When [port power is Coupled](#), setting port power for one port will apply port power for all source ports.

All subsequent segments that are added assume the new Power Level value.

Valid either for Source ranges or for Primary range when [coupled](#) to the source.

Parameters

- <num> Any existing channel number; if unspecified, value is set to 1.
- <n> Range number. If unspecified, value is set to 1.
- <s> Segment number to modify. Choose any existing segment number.
- <p> Port number of the source. If unspecified, value is set to 1.
- <num> Power level in dBm.

Note: The range of settable power values depends on the PNA model and if source attenuators are installed. To determine the range of values, send SOUR:POW? MAX and SOUR:POW? MIN. ([SOUR:POW:ATT:AUTO](#) must be set to ON).

Actual achievable leveled power depends on frequency.

Examples

```
SENS:FOM:RANG:SEGM:POW -5
```

```
sense2:fom:range2:segment4:power2:level 5
```

Query Syntax SENSE<cnum>:FOM:RANGe<n>:SEGMent<s>:POWer<p>[:LEVel]?

Return Type Numeric

Default 0

SENSe<cnum>:FOM:RANGe<n>:SEGMent:POWer[:LEVel]:CONTRol <bool>

(Read-Write) Specifies whether Power Level is to be set independently for each segment.

Valid either for Source ranges or for Primary range. Primary range value is ignored unless Source is [coupled](#) to Primary.

Parameters

- <cnum> Any existing channel number; if unspecified, value is set to 1.
- <n> Range number. If unspecified, value is set to 1.
- <bool> ON (or 1) - Power level will be set for each segment.
OFF (or 0) - Use the channel power level setting.

Examples

```
SENS:FOM:RANG:SEGM:POW:CONT 0  
sense2:fom:range2:segment:power:control on
```

Query Syntax SENSE<cnum>:FOM:RANGe<n>:SEGMENT:POWer[:LEVeL]:CONTrol?

Return Type Boolean

Default OFF (or 0)

SENSe<cnum>:FOM:RANGe<n>:SEGMENT<s>[:STATe] <bool>

(Read-Write) Turns the specified sweep segment ON or OFF.

Parameters

- <cnum> Any existing channel number; if unspecified, value is set to 1.
- <n> Range number. If unspecified, value is set to 1.
- <s> Segment number to be turned ON or OFF. Choose any existing segment number.
- <bool> ON (or 1) - turns segment ON.
OFF (or 0) - turns segment OFF.

Examples

```
SENS:FOM:RANG:SEGM 0  
sense2:fom:range2:segment4:state on
```

Query Syntax SENSE<cnum>:FOM:RANGe<n>:SEGMENT<s>[STATe]?

Return Type Boolean

Default OFF (or 0)

SENSe<cnum>:FOM:RANGe<n>:SEGMENT<s>:SWEep:POINts <num>

(Read-Write) Sets the number of data points for the specified sweep segment.

Parameters

- <cnm> Any existing channel number; if unspecified, value is set to 1.
- <n> Range number. If unspecified, value is set to 1.
- <s> Segment number to modify. Choose any existing segment number.
- <num> Number of points in the segment. The total number of points in all segments cannot exceed 20001. A segment can have as few as 1 point.

Examples

```
SENS:FOM:RANG:SEGM:SWE:POIN 101  
sense2:fom:range2:segment4:sweep:points 201
```

Query Syntax SENSE<cnm>:FOM:RANGe<n>:SEGMent<s>:SWEep:POINts?

Return Type Numeric

Default 21

SENSe<cnm>:FOM:RANGe<n>:SEGMent<s>:SWEep:TIME <num>

(Read-Write) Sets the time the PNA takes to sweep the specified segment.

Valid ONLY for receiver ranges.

Parameters

- <cnm> Any existing channel number; if unspecified, value is set to 1.
- <n> Range number. If unspecified, value is set to 1.
- <s> Segment number for which to set sweep time.
- <num> Sweep time in seconds. Choose a number between 0 and 100

Examples

```
SENS:FOM:RANG:SEGM:SWE:TIME 1  
sense2:fom:range2:segment3:sweep:time .1
```

Query Syntax SENSE<cnm>:FOM:RANGe<n>:SEGMent<s>:SWEep:TIME?

Return Type Numeric

Default Not Applicable

SENSe<cnm>:FOM:RANGe<n>:SEGMent:SWEep:TIME:CONTrol <bool>

(Read-Write) Specifies whether Sweep Time can be set independently for each sweep segment.

Valid either for Receiver ranges or for Primary range. Primary range value is ignored unless Receiver is [coupled](#) to Primary.

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1.

<n> Range number. If unspecified, value is set to 1.

<bool> **ON** (or 1) - Sweep time will be set for each segment.

OFF (or 0) - Use the channel sweep time setting.

Examples

```
SENS:FOM:RANG:SEGM:SWE:TIME:CONT 1
```

```
sense2:fom:range2:segment:sweep:time:control off
```

Query Syntax SENSE<cnum>:FOM:RANGe<n>:SEGMENT:SWEep:TIME:CONTROL?

Return Type Boolean

Default OFF

Last Modified:

21-Jun-2007 Increased max number of points

Sense:Frequency Commands

Sets the frequency sweep functions of the analyzer.



Click on a [blue](#) keyword to view the command details.

See Also

- [Example](#) using some of these commands.
- [Learn about Frequency Sweep](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

SENSe<num>:FREQUency:CENTer <num>

(Read-Write) Sets the center frequency of the analyzer.

Parameters

<num> Any existing channel number. If unspecified, value is set to 1

<num> Center frequency. Choose any number between the **minimum** and **maximum** frequency limits of the analyzer. Units are Hz.

This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
SENS:FREQ:CENT 1000000
```

```
sense2:frequency:center 1mhz
```

```
sense2:frequency:center 1e6
```

Query Syntax SENSe<num>:FREQUency:CENTer?

Return Type Numeric

Default Center of the analyzer's frequency span

SENSe<num>:FREQUency[:CW |:FIXed] <num>

(Read-Write) Sets the Continuous Wave (or Fixed) frequency. Must also send [SENS:SWEEP:TYPE CW](#) to put the analyzer into CW sweep mode.

Parameters

- <num> Any existing channel number. If unspecified, value is set to 1
- <num> CW frequency. Choose any number between the **minimum** and **maximum** frequency limits of the analyzer. Units are Hz.

This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
SENS:FREQ 1000000
SENS:FREQ:CW MIN
sense2:frequency:fixed 1mhz
```

Query Syntax SENSE<num>:FREQUENCY[:CW | :FIXed]?

Return Type Numeric

Default 1 GHz

SENSe<num>:FREQUENCY:SPAN <num>

(Read-Write) Sets the frequency span of the analyzer.

Parameters

- <num> Any existing channel number. If unspecified, value is set to 1
- <num> Frequency span in Hz. Choose any number between **0** (minimum) and the **maximum** frequency span of the analyzer.

This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
SENS:FREQ:SPAN 1000000
sense2:frequency:span max
```

Query Syntax SENSe<num>:FREQUENCY:SPAN?

Return Type Numeric

Default Maximum frequency span of the analyzer

SENSe<num>:FREQUENCY:START <num>

(Read-Write) Sets the start frequency of the analyzer.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <num> Start frequency. Choose any number between the **MIN** and **MAX** frequency limits of the analyzer. Units are Hz

If **FREQ:START** is set greater than **FREQ:STOP**, then **STOP** is set equal to **START**.

This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
SENS:FREQ:STAR 1000000  
sense2:frequency:start MIN
```

Query Syntax SENSE<cnum>:FREQUENCY:START?

If [Sweep type is segment](#), this query returns the start frequency of the first segment.

Return Type Numeric

Default Minimum frequency of the analyzer

SENSe<cnum>:FREQUENCY:STOP <num>

(Read-Write) Sets the stop frequency of the analyzer.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <num> Stop frequency. Choose any number between: the **minimum** and **maximum** frequency limits of the analyzer. Units are Hz.

If **FREQ:STOP** is set less than **FREQ:START**, then **START** will be set equal to **STOP**.

This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
SENS:FREQ:STOP 1000000  
sense2:frequency:stop max
```

Query Syntax SENSe<cnum>:FREQUENCY:STOP?

If [Sweep type is segment](#), this query returns the stop frequency of the last segment.

Return Type Numeric

Default Maximum frequency of the analyzer

Last Modified:

5-Feb-2008 Modified Max and Min notes

SENSe:GCSetup Commands

Controls the Gain Compression configuration.

SENSe:GCSetup:

AMODe <char>

COMPression:

| **ALGorithm** <char>

| **BACKoff:LEVel** <num>

| **DELTA:X** <num>

| **DELTA:Y** <num>

| **INTerpolation** <bool>

| **LEVel** <num>

| **SATuration:LEVel**

EOSoperation <string>

PMAP

| **INPut?**

| **OUTPut?**

POWer:

| **LINear:INPut:LEVel** <num>

| **REVerse:LEVel** <num>

| **STARt:LEVel** <num>

| **STOP:LEVel** <num>

SAFE:

| **CPADjustment** <num>

| **ENABLE** <bool>

| **FPADjustment** <num>

| **FTHReshold** <num>

| **MLIMit** <num>

SFA?

SMART:

| [MITerations <num>](#)

| [SITerations <bool>](#)

| [STIMe <num>](#)

| [TOLerance <num>](#)

SWEp:

| [FREQuency:POINts <num>](#)

| [POWer:POINts <num>](#)

Click on a [blue](#) keyword to view the command details.

See Also

Other Gain Compression commands

- [CALC:CUSTom:DEFine](#) - creates a gain compression measurement.
- [CALC:GCMeas:ANAL](#) - Gain Compression Analysis settings
- **GCA Calibration** uses the [Guided Calibration commands](#), except for the following:
 - [Sens:Corr:GCS:Power](#) - sets power level for Source Power Cal

GCX

- Setup Mixer using [Sense:Mixer commands](#)
 - Calibrate using [SMC commands](#) and [Guided commands](#)
 - **Example Programs**
 - [Create and Cal a Gain Compression Measurement](#)
 - [Create and Cal a GCX Measurement](#)
 - [Learn about Gain Compression Application](#)
 - [Synchronizing the PNA and Controller](#)
 - [SCPI Command Tree](#)
-

SENSe<ch>:GCSetup:AMODE <char>

(Read-Write) Set and read the method by which gain compression data is acquired.

Parameters

- <ch> Any existing GCA channel. If unspecified, value is set to 1.
- <char> Choose from:
- PFREQuency - 2D Power Per Frequency
 - FPOWer - 2D Frequency Per Power
 - SMARTsweep - Smart Sweep

Examples

```
SENS:GCS:AMOD SMAR  
sense:gcsetup:amode pfrequency
```

Query Syntax SENSE<ch>:GCSetup:AMODE ?

Return Type Character

Default SMARTsweep

SENSe<ch>:GCSetup:COMPression:ALGorithm <char>

(Read-Write) Set and read the algorithm method used to compute gain compression.

Parameters

- <ch> Any existing GCA channel. If unspecified, value is set to 1.
- <char> Algorithm method. Choose from:
- CFLG - compression from linear gain
 - CFMG - compression from maximum gain
 - BACKoff - compression from BackOff
 - XYCOM - X/Y Compression
 - SAT - compression from saturation

Examples

```
SENS:GCS:COMP:ALG BACK  
sense:gcsetup:compression:algorithm XYcom
```

Query Syntax SENSE<ch>:GCSetup:COMPression:ALGorithm?

Return Type Character

Default CFLG

SENSE<ch>:GCSetup:COMPression:BACKoff:LEVel <num>

(Read-Write) Set and read value for the BackOff compression algorithm.

Parameters

- <ch> Any existing GCA channel. If unspecified, value is set to 1.
- <num> Backoff value in dB. Choose from 30 to (-30)

Examples

```
SENS:GCS:COMP:BACK:LEV 10  
sense:gcsetup:compression:backoff:level 5
```

Query Syntax SENSE<ch>:GCSetup:COMPression:BACKoff:LEVel?

Return Type Numeric

Default 10

SENSE<ch>:GCSetup:COMPression:DELTA:X <num>

(Read-Write) Set and read the 'X' value in the delta X/Y compression algorithm.

Parameters

- <ch> Any existing GCA channel. If unspecified, value is set to 1.
- <num> X value in dB. Choose from 30 to (-30)

Examples

```
SENS:GCS:COMP:DELTA:X 9  
sense:gcsetup:compression:delta:X 8
```

Query Syntax SENSE<ch>:GCSetup:COMPression:DELTA:X?

Return Type Numeric

Default 10

SENSE<ch>:GCSetup:COMPression:DELTA:Y <num>

(Read-Write) Set and read the 'Y' value in the delta X/Y compression algorithm.

Parameters

- <ch> Any existing GCA channel. If unspecified, value is set to 1.
- <num> Y value in dB. Choose from 30 to (-30)

Examples

```
SENS:GCS:COMP:DELTA:Y 9  
sense:gcsetup:compression:delta:Y 8
```

Query Syntax SENSE<ch>:GCSetup:COMPression:DELTA:Y?

Return Type Numeric

Default 9

SENSE<ch>:GCSetup:COMPression:INTerpolation <bool>

(Read-Write) Sets whether or not interpolation should be performed on 2D measured compression data. Applies ONLY to [2D acquisition modes](#).

Parameters

- <ch> Any existing GCA channel. If unspecified, value is set to 1.
- <bool> Choose from:
 - ON** or (**1**) Interpolate the results
 - OFF** or (**0**) Do NOT interpolate the results but return the value closest to compression.

Examples

```
SENS:GCS:COMP:INT 1  
sense:gcsetup:compression:interpolation off
```

Query Syntax SENSE<ch>:GCSetup:COMPression:INTerpolation?

Return Type Boolean

Default OFF

SENSE<ch>:GCSetup:COMPression:LEVel <num>

(Read-Write) Set and read the desired gain reduction (from reference gain).

This value is used for Compression from Linear Gain and Compression from Maximum Gain.

Use [SENS:GCS:COMP:ALG CFLG](#) to set this compression algorithm.

Parameters

- <ch> Any existing GCA channel. If unspecified, value is set to 1.
- <num> Compression level in dB. Choose a value greater than 0.1 dB.

Examples

```
SENS:GCS:COMP:LEV 1  
sense:gcsetup:compression:level 3
```

Query Syntax SENSE<ch>:GCSetup:COMPression:LEVel?

Return Type Numeric

Default 1

SENSE<ch>:GCSetup:COMPression:SATuration:LEVel <num>

(Read-Write) Set and read the deviation dB from the maximum Pout. This is the point of saturation.

Use [SENS:GCS:COMP:ALG CFLG](#) to set this compression algorithm.

Parameters

- <ch> Any existing GCA channel. If unspecified, value is set to 1.
- <num> Saturation level in dB. Choose a value greater than 0.01 dB.

Examples

```
SENS:GCS:COMP:SAT:LEV 1  
sense:gcsetup:compression:saturation:level 3
```

Query Syntax SENSE<ch>:GCSetup:COMPressionSATuration:LEVel?

Return Type Numeric

Default .1 dB

SENSE<ch>:GCSetup:EOSoperation <char>

(Read-Write) Set and read the This setting is used to protect a sensitive device from too much power during the sweep retrace. Other instrument settings or channels may over-ride this setting. [Learn more.](#)

Parameters

- <ch> Any existing GCA channel. If unspecified, value is set to 1.
- <char> End Of Sweep operation. Choose from:
- STANdard Use the default PNA method. [Learn more.](#)
 - POFf Always turn power OFF while waiting.
 - PSTArt Sweep Start power
 - PSTOp Sweep Stop power.

Examples

```
SENS:GCS:EOS PSTA  
sense:gcsetup:eosoperation standard
```

Query Syntax SENSE<ch>:GCSetup:EOSoperation?

Return Type Character

Default STANdard

SENSe<ch>:GCSetup:PMAP <in>,<out>

(Write-only) Set the DUT-to-PNA port mapping for the Gain Compression measurement.

Parameters

- <ch> Any existing GCA channel. If unspecified, value is set to 1.
- <in> PNA port which is connected to the DUT input.
- <out> PNA port which is connected to the DUT output.

Examples

```
SENS:GCS:PMAP 1,2  
sense:gcsetup:pmap 2,1
```

Query Syntax Not Applicable

Default 1,2

SENSe<ch>:GCSetup:PMAP:INPut?

(Read-only) Read the PNA port number to be connected to the DUT Input.

Use [SENS:GCS:PORTMap](#) to set the port mapping.

Parameters

<ch> Any existing GCA channel. If unspecified, value is set to 1.

Examples

```
SENS:GCS:PMAP:INP?  
sense:gcsetup:pmap:input?
```

Return Type Numeric

Default 1

SENSe<ch>:GCSetup:PMAP:OUTPut?

(Read-only) Read the PNA port number to be connected to the DUT Output.

Parameters

<ch> Any existing GCA channel. If unspecified, value is set to 1.

Examples

```
SENS:GCS:PMAP:OUTP?  
sense:gcsetup:pmap:output?
```

Return Type Numeric

Default 2

SENSe<ch>:GCSetup:POWER:LINear:INPut:LEVel <num>

(Read-Write) Set and read the input power at which Linear Gain and all S-parameters are measured.

Parameters

<ch> Any existing GCA channel. If unspecified, value is set to 1.

<num> Input power level in dBm. Choose a value from +30 to (-30).

Examples

```
SENS:GCS:POW:LIN:INP:LEV 0  
sense:gcsetup:power:linear:input:level -10
```

Query Syntax SENSe<ch>:GCSetup:POWER:LINear:INPut:LEVel?

Return Type Numeric

Default -25 dBm

SENSE<ch>:GCSetup:POWER:REVerse:LEVel <num>

(Read-Write) Set and read the reverse power level to the DUT. This is applied to the DUT output port when making reverse measurements like S22.

Parameters

- <ch> Any existing GCA channel. If unspecified, value is set to 1.
<num> Reverse power level in dBm. Choose a value from +30 to (-30).

Examples

```
SENS:GCS:POW:REV:LEV 0  
sense:gcsetup:power:reverse:level -5
```

Query Syntax SENSE<ch>:GCSetup:POWER:REVerse:LEVel?

Return Type Numeric

Default -5

SENSE<ch>:GCSetup:POWER:STARt:LEVel <num>

(Read-Write) Set and read the start power level.

Parameters

- <ch> Any existing GCA channel. If unspecified, value is set to 1.
<num> Start power level in dBm. Choose a value from +30 to (-30).

Examples

```
SENS:GCS:POW:STAR:LEV 0  
sense:gcsetup:power:start:level -5
```

Query Syntax SENSE<ch>:GCSetup:POWER:STARt:LEVel?

Return Type Numeric

Default -30

SENSE<ch>:GCSetup:POWER:STOP:LEVel <num>

(Read-Write) Set and read the stop power level.

Parameters

- <ch> Any existing GCA channel. If unspecified, value is set to 1.
- <num> Stop power level in dBm. Choose a value from +30 to (-30).

Examples

```
SENS:GCS:POW:STOP:LEV 0  
sense:gcsetup:power:stop:level -5
```

Query Syntax SENSE<ch>:GCSetup:POWER:STOP:LEVel?

Return Type Numeric

Default -5

SENSE<ch>:GCSetup:SAFE:CPADjustment <num>

(Read-Write) Set and read the Safe Sweep COARSE power adjustment. [Learn more.](#)

Parameters

- <ch> Any existing GCA channel. If unspecified, value is set to 1.
- <num> Coarse power adjustment setting in dBm. Choose a value from +30 to (-30).

Examples

```
SENS:GCS:SAFE:CPAD 2  
sense:gcsetup:safe:cpadjustment 3.5
```

Query Syntax SENSE<ch>:GCSetup:SAFE:CPADjustment?

Return Type Numeric

Default 3.0

SENSE<ch>:GCSetup:SAFE:ENABLE <bool>

(Read-Write) Set and read the (ON | OFF) state of Safe Sweep mode. [Learn more](#)

Parameters

- <ch> Any existing GCA channel. If unspecified, value is set to 1.
- <num> **(Boolean)** - Safe Sweep state. Choose from:
 - OFF** (or **0**) - Disable Safe Sweep
 - ON** (or **1**) - Enable Safe Sweep

Examples

```
SENS:GCS:SAFE:ENAB 0  
sense:gcsetup:safe:enable 1
```

Query Syntax SENSE<ch>:GCSetup:SAFE:ENABLE?

Return Type Boolean

Default 0

SENSe<ch>:GCSetup:SAFE:FPADjustment <num>

(Read-Write) Set and read the Safe Sweep FINE power adjustment. [Learn more](#)

Parameters

- <ch> Any existing GCA channel. If unspecified, value is set to 1.
- <num> Fine power adjustment setting in dBm. Choose a value from +30 to (-30).

Examples

```
SENS:GCS:SAFE:FPAD 2  
sense:gcsetup:safe:fpadjustment .5
```

Query Syntax SENSe<ch>:GCSetup:SAFE:FPADjustment?

Return Type Numeric

Default 1.0 dBm

SENSe<ch>:GCSetup:SAFE:FTHReshold <num>

(Read-Write) Set and read the compression level in which Safe Sweep changes from the COARSE power adjustment to the FINE power adjustment. [Learn more](#)

Parameters

- <ch> Any existing GCA channel. If unspecified, value is set to 1.
- <num> Threshold setting in dB. Choose a value from +30 to (-30).

Examples

```
SENS:GCS:SAFE:FTHR .1  
sense:gcsetup:safe:fthreshold .75
```

Query Syntax SENSE<ch>:GCSetup:SAFE:FTHReshold?

Return Type Numeric

Default 0.5 dB

SENSE<ch>:GCSetup:SAFE:MLimit <num>

(Read-Write) When the PNA port that is connected to the DUT Output measures this value, the input power to the DUT is no longer incremented at that frequency. Safe Mode must be enabled with [SENS:GCS:SAFE:ENAB ON](#) [Learn more](#)

Parameters

- <ch> Any existing GCA channel. If unspecified, value is set to 1.
- <num> Maximum power limit in dBm. Choose a value from -100 to +100

Examples

```
SENS:GCS:SAFE:MLIM 20  
sense:gcsetup:safe:mlimit 30
```

Query Syntax SENSE<ch>:GCSetup:SAFE:MLIMit?

Return Type Numeric

Default 30

SENSE<ch>:GCSetup:SFAilures?

(Read-only) Returns a comma-separated list of the frequency indexes that were out of tolerance for SMART Sweep mode, or at the power limit for 2D acquisition modes. Zero (0) is the first frequency data point.

Must be Single triggered. Invalid results occur if the GCA channel is continuously sweeping.

Parameters

<ch> Any existing GCA channel. If unspecified, value is set to 1.

Examples

```
SENS:GCS:SFA?  
  
sense:gcsetup:sfailures?
```

Return Type Comma-separated list of frequency indexes.

Default Not Applicable

SENSe<ch>:GCSetup:SMARt:MITerations <num>

(Read-Write) Set and read the maximum permitted number of iterations which SMART Sweep may utilize to find the desired compression level, to within the specified tolerance.

Parameters

<ch> Any existing GCA channel. If unspecified, value is set to 1.

<num> Maximum number of iterations. Choose a value between 1 and 10e8

Examples

```
SENS:GCS:SMAR:MIT 5  
  
sense:gcsetup:smart:miterations 3
```

Query Syntax SENSe<ch>:GCSetup:SMARt:MITerations?

Return Type Numeric

Default 20

SENSe<ch>:GCSetup:SMARt:SITerations <bool>

(Read-Write) Set and read enable for showing intermediate results for each iteration of SMART Sweep

Parameters

- <ch> Any existing GCA channel. If unspecified, value is set to 1.
- <bool> Choose from:
- ON** or **(1)** Compression traces are updated after each iteration.
 - OFF** or **(0)** Compression traces are updated after ALL iterations are complete.

Examples

```
SENS:GCS:SMAR:SIT 1
sense:gcsetup:smart:siterations off
```

Query Syntax SENSE<ch>:GCSetup:SMART:SIterations?

Return Type Boolean

Default OFF

SENSE<ch>:GCSetup:SMART:STIME <num>

(Read-Write) Set and read the amount of time SMART Sweep will dwell at the first point where the input power changes by the Backoff or X level. Applies only to SMART Sweep when Backoff or XY compression methods are selected. [Learn more.](#)

Parameters

- <ch> Any existing GCA channel. If unspecified, value is set to 1.
- <bool> Settling time in seconds. Choose any positive value.

Examples

```
SENS:GCS:SMAR:STIM 1
sense:gcsetup:smart:stime .1
```

Query Syntax SENSE<ch>:GCSetup:SMART:STIME?

Return Type Numeric

Default 0

SENSE<ch>:GCSetup:SMART:TOLerance <num>

(Read-Write) Set and read the acceptable range SMART Sweep will allow for the measured compression level.

Parameters

- <ch> Any existing GCA channel. If unspecified, value is set to 1.
- <num> Tolerance level in dBm. Choose a value between .01 and 10

Examples

```
SENS:GCS:SMAR:TOL .1  
sense:gcsetup:smart:tolerance .05
```

Query Syntax SENSE<ch>:GCSetup:SMART:TOLerance?

Return Type Numeric

Default .05

SENSE<ch>:GCSetup:SWEep:FREQuency:POINts <num>

(Read-Write) Set and read the number of data points in each frequency sweep. [Learn more](#)

Parameters

- <ch> Any existing GCA channel. If unspecified, value is set to 1.
- <num> Frequency points. Do not exceed the max number of data points.

[See Data Points Limit](#)

Examples

```
SENS:GCS:SWE:FREQ:POIN 201  
sense:gcsetup:sweep:frequency:points 101
```

Query Syntax SENSE<ch>:GCSetup:SWEep:FREQuency:POINts?

Return Type Numeric

Default 201

SENSE<ch>:GCSetup:SWEep:POWER:POINts <num>

(Read-Write) Set and read the number of data points in each power sweep. Applies ONLY to 2D [acquisition modes](#).

Parameters

- <ch> Any existing GCA channel. If unspecified, value is set to 1.
- <num> Power points. Do not exceed the max number of data points.

[See Data Points Limit](#)

Examples

```
SENS:GCS:SWE:POW:POIN 50  
sense:gcsetup:sweep:power:points 21
```

Query Syntax SENSE<ch>:GCSetup:SWEp:POWer:POINTs?

Return Type Numeric

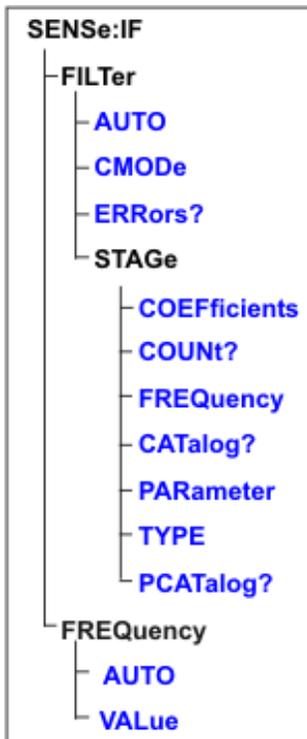
Default 26

Last Modified:

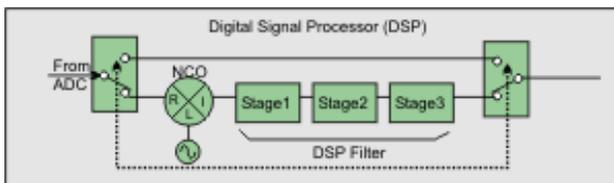
- 6-Oct-2011 Added GCX notes
- 11-Jan-2011 Minor edits
- 9-Nov-2007 MX New topic

Sense:IF Commands

Controls the DSP filters for use with the PNA X.



- Click on a [blue](#) keyword to view the command details.
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)
- [Pulse commands](#)



PNA-X DSP Block diagram

All of the Sense:IF commands, except the frequency commands, make settings related to the DSP section of the IF Path.

See the entire [IF Path Configuration](#).

- For any of the Filter "Stage" parameters to take effect, [SENS:IF:FILT:AUTO](#) must be set to OFF (MANUAL) and mode, and [SENS:IF:FILT:CMOD](#) must be set to OFF.
- Stage2 settings are ignored when using DSP 5 versions.
- Programs that control these settings, or state files that are saved, will yield different results when run or recalled on PNAs with DSP 4 versions versus DSP 5 versions. [Learn more about DSP versions.](#)
- **Critical Note:** These commands act on the selected measurement. You can select one measurement for each channel using [Calc:Par>Select](#)

SENSe<cnum>:IF:FILTer:AUTO <bool>

(Read-Write) Sets and returns whether the PNA configures the 3-stage digital filter settings or they will be configured manually. When making manual settings, also send [SENS:IF:FILT:CMOD OFF](#) which routes the IF through the 3-stage filter.

Critical Note

Parameters

<cnum> Existing channel number. If unspecified, <cnum> is set to 1.

<bool> (Boolean)

ON (or **1**) - Automatic: PNA controls digital filter settings.

OFF (or **0**) - Manual: You control digital filter settings using other Sens:IF commands.

Examples

```
SENS:IF:FILT:AUTO 1
sense2:if:filter:auto 0
```

Query Syntax SENSe<cnum>:IF:FILTer:AUTO?

Return Type Boolean

Default ON

SENSe:IF:FILTer:CMODE <bool>

(Read-Write) Sets and returns the ADC capture mode modeled as a 2-pole switch in the above diagram. The switch either bypasses or routes the IF through the 3-stage digital filter.

Critical Note

Parameters

<bool> (Boolean)

ON (or 1) - The digital filters are bypassed and the raw ADC readings are taken directly. With DSP 4 versions, a maximum of 4096 data points per sweep can be acquired.

With DSP 5 versions, the [PNA maximum data points](#) per sweep can be acquired.

[Learn more about DSP Versions.](#)

OFF (or 0) - The digital filters are used to process IF information. The filters can be configured automatically or manually using [SENS:IF:FILT:AUTO](#).

Examples

```
SENS:IF:FILT:CMOD 1  
sense2:if:filter:cmode 0
```

Query Syntax

SENSe<cnum>:IF:FILT:CMODE?

Return Type

Boolean

Default

OFF

SENSe<cnum>:IF:FILT:ERRors?

(Read-only) Returns the error string associated with the digital filters. The return string has three fields separated by commas: "stage1 status, stage2 status, stage3 status"

Each of these fields can contain one or more of the following error codes:

- **NO ERROR**
- ***NUMBER-OF-COEFFICIENTS** - the number of coefficients is excessive for that filter-stage
- ***COEFFICIENT VALUE** - one or more coefficients are out of range for that filter-stage
- ***SUM-OF-COEFFICIENTS** - the sum of all coefficients is excessive for that filter-stage,
- ***FREQUENCY** - the frequency for Stage 1 is out of range (only applies stage1 field),
- ***PARAMETER** - one or more parameters are out of range (only applies to stage 3 field)

Critical Note

Parameters

<cnum> Existing channel number. If unspecified, <cnum> is set to 1.

Examples

```
SENS:IF:FILT:ERR?  
  
'example return strings"  
  
NO ERROR, NO ERROR, NO ERROR  
indicates no errors,  
  
*SUM-OF-COEFFICIENTS, NO ERROR, NO ERROR  
indicates that the sum of all filter coefficients exceed the  
maximum value for the Stage-1 filter,  
  
*COEFFICIENT *SUM-OF-COEFFICIENTS, NO ERROR, *PARAMETER  
indicates a problems with Stage 1 coefficients and a problem with  
one or more of the parameters associated with the Stage 3 filter.
```

Return Type

String

Default

Not Applicable

SENSe<cnum>:IF:FILTER:STAGe<n>:COEFFicients <coef>

(Read-Write) Sets and returns the digital filter coefficients of the specified stage.

Note: Stage2 settings are ignored when using DSP Version 5. [Learn more.](#)

Critical Note

Parameters

<cnum> Existing channel number. If unspecified, <cnum> is set to 1.

<n> Stage number. Choose 1, 2, or 3.

<coef> Filter coefficients.

For stages 1 and 2, values can be sent as integers or in floating point format. If floating point, the values are truncated to integers. The valid range is 0 to 131071.

For stage 3: An array of floating point values. The valid range is any value within range of a floating point.

Examples

```
SENS:IF:FILT:STAG2:COEF 0,0.1,0.7,0.7,0.1  
  
sense2:if:filter:stage3:coefficients  
+0.0E+000,+6.4E+001,+2.56E+002
```

Query Syntax

SENSe<cnum>:IF:FILTER:STAGe<n>:COEFFicients?

Example

```
SENS:IF:FILT:STAG2:COEF?
```

Return Type

Floating point values

Default Stage dependent

SENSe<cnum>:IF:FILTer:STAGe<n>:COUNT? [char]

(Read-only) Returns the number of taps in the digital filter of the specified stage. The filter sample count setting is only used when [SENSe:IF:FILTer:AUTO](#) is set to False (MANUAL).

Note: Stage2 settings are ignored when using DSP Version 5. [Learn more.](#)

Critical Note

Parameters

<cnum> Existing channel number. If unspecified, <cnum> is set to 1.

<n> Stage number. Choose 1, 2, or 3

[char] Optional parameter. Choose from:

MIN - returns the minimum number of coefficients for the specified stage. Stage1: **10**, Stage2: **1**, Stage3: **2**

MAX - returns the maximum number of coefficients for the specified stage. Stage1 & 2: **1024**, Stage3: **102400**

Examples

```
SENS:IF:FILT:STAG2:COUN?
```

```
sense2:if:filter:stage1:count? max
```

Return Type

Numeric

Default Stage dependent

SENSe<cnum>:IF:FILTer:STAGe1:FREQuency <value>

(Read-Write) Sets and returns the Numerically Controlled Oscillator (NCO) frequency. This command is only used when [SENSe:IF:FILTer:AUTO](#) is set to False (Manual).

Critical Note

Parameters

<cnum> Existing channel number. If unspecified, <cnum> is set to 1.

<value> Stage 1 Frequency. Min value= 0 Hz

With DSP 4 versions, Max value= 15 MHz.

With DSP 5 versions, Max value = 38 MHz.

Or programmatically use the Max and Min queries to determine the range of settable values.

[Learn more about DSP versions.](#)

Examples

```
SENSe:IF:FILTer:STAGe1:FREQ 9e6  
sense2:if:filter:stage1:frequency 9.2e6
```

Query Syntax

SENSe<cnum>:IF:FILTer:STAGe1:FREQuency?

'returns the current parameter value

SENSe<cnum>:IF:FILTer:STAGe1:FREQuency?Min

'returns the minimum frequency value.

SENSe<cnum>:IF:FILTer:STAGe1:FREQuency?Max

'returns the maximum frequency value.

Return Type

Numeric

Default

Nominal IF Frequency. [Learn more](#)

SENSe<cnum>:IF:FILTer:STAGe3:CATalog?

(Read-only) Returns a list of strings for the currently supported filter types that can be used for the stage 3 filter. This command is only used when [SENSe:IF:FILTer:AUTO](#) is set to False (Manual). See [SENS:IF:FILT:STAGe3:TYPE](#) for a list of currently supported filter types.

Critical Note

Parameters

<num> Existing channel number. If unspecified, <num> is set to 1.

Examples

```
SENS:IF:FILT:STAGe3:CAT?  
sense2:if:filter:stage3:catalog?
```

Return Type

String array

Default

Not Applicable

SENSe<num>:IF:FILTer:STAGe3:PARAmeter <p>, <value>

(Read-Write) Sets and returns the Stage 3 filter parameters.

Must first select the filter type ([SENS:IF:FILT:STAGe3:TYPE](#)) before setting these parameters

Use [SENSe:IF:FILT:STAGe3:PCAT?](#) to return a list of the available parameters for the currently selected filter type.

Critical Note

Parameters

<num> Existing channel number. If unspecified, <num> is set to 1.

<p> (String) Filter parameter. Case Sensitive. Choose from:

"C" - Tap count (Tukey, RECT, PWIN)

"P" - Period (PWIN ONLY)

"D" - Delay (PWIN ONLY)

"W" - Width (PWIN ONLY)

"R" - Ramp Count (PWIN ONLY)

"M" - Number of times to repeat the user-supplied array for each data point (COEF ONLY)

<value> (Numeric) Parameter Value for the specified stage 3 parameter. Use the query form to return the minimum and maximum values for the specified parameter.

Examples

```
SENS:IF:FILT:STAGe3:PAR "C",64  
sense2:if:filter:stage3:parameter "d",0.5E-6
```

Query Syntax SENSE<cnum>:IF:FILTer:STAGe3:PARAmeter? <p>
returns the current parameter value

SENSe<cnum>:IF:FILTer:STAGe3:PARAmeter? <p>,Min
returns the minimum parameter value.

SENSe<cnum>:IF:FILTer:STAGe3:PARAmeter? <p>,Max
returns the maximum parameter value.

Examples

```
SENS:IF:FILT:STAGe3:PAR? "C"  
sense2:if:filter:stage3:parameter? "d",min
```

Return Type Numeric

Default

RECT: C = 1

PWIN: C=1E6, P=10ms, D=50us, W=50us, R=7

TUKEY: C=1

COEF: M=1

SENSe<cnum>:IF:FILTer:STAGe3:PCATalog?

(Read-only) Returns a list of the available parameters for the currently selected filter type.

Critical Note

Parameters

<cnum> Existing channel number. If unspecified, <cnum> is set to 1.

Examples

```
SENS:IF:FILT:STAGe3:PCAT?  
sense2:if:filter:stage3:pcatalog?
```

Return Type String

Default

Not Applicable

SENSe<cnum>:IF:FILTer:STAGe3:TYPE <value>

(Read-Write) Sets and returns the Stage 3 filter type. This command is only used when [SENSe:IF:FILTer:AUTO](#) is set to False (Manual).

Critical Note

Parameters

<num> Existing channel number. If unspecified, <num> is set to 1.

<value> (String) Filter type. Chose from:

- **"RECT"** Rectangular Window Filter
- **"TUKEY"** Tukey Filter
- **"PWIN"** Pulse Window Filter
- **"COEF"** User-supplied array

Examples

```
SENS:IF:FILT:STAGe3:TYPE RECT
```

```
sense2:if:filter:stage3:type pwin
```

Query Syntax

```
SENSe<num>:IF:FILTer:STAGe3:TYPE?
```

Return Type

String

Default

"TUKEY"

SENSe<num>:IF:FREQuency:AUTO <bool>

(Read-Write) Sets and returns the method for specifying the way the IF Frequency is determined.

Critical Note

Parameters

<num> Existing channel number. If unspecified, <num> is set to 1.

<bool> (Boolean)

ON (or **1**) - Automatic. PNA determines the setting for the IF frequency. The IF frequency is based on many PNA settings, including measurement frequency. Therefore, it is NOT possible to read the IF frequency that is being used.

OFF (or **0**) - Manual. Use [SENS:IF:FREQ](#) to set the frequency.

Examples

```
SENS:IF:FREQ:AUTO 1
```

```
sense2:if:frequency:auto 0
```

Query Syntax

```
SENSe<num>:IF:FREQuency:AUTO?
```

Return Type

Boolean

Default

ON

SENSe<cnum>:IF:FREQuency[:VALue] <value>

(Read-Write) Sets and returns the IF frequency for ALL receiver paths being used for the specified channel. To set this frequency, [SENS:IF:FREQ:AUTO](#) must be set to OFF (Manual). Also returns the maximum and minimum allowable frequency settings. [Learn more.](#)

Critical Note

Parameters

- <cnum> Existing channel number. If unspecified, <cnum> is set to 1.
- <value> (Numeric) Frequency value. Use the Max and Min Queries to determine the range of this setting. (SENS:IF:FREQ? Max)

Examples

```
SENS:IF:FREQ 9.1e6  
sense2:if:frequency:value 8.9e6
```

Query Syntax

SENSe<cnum>:IF:FREQuency?

'returns the current frequency setting

SENSe<cnum>:IF:FREQuency? Max

'returns the maximum allowable frequency setting

SENSe<cnum>:IF:FREQuency? Min

'returns the minimum allowable frequency setting

Return Type

Numeric

Default

9 MHz

Last Modified:

- | | |
|-------------|---------------------------------|
| 29-Aug-2013 | Fixed several typos |
| 7-Feb-2013 | Added stage 3 parameters (9.90) |
| 26-May-2011 | Changed IF freq default |
| 26-Aug-2010 | Updated for DSP 5 |
| 18-Jan-2007 | MX New topic |

Sense:IMD Commands

Controls an IMD or IMDx measurement configuration.

SENSe:IMD

SWEEp:TYPE

CSO:

| **NDP**roducts

| **NOR**Malized:POWer

| **OFF**Set

CTB

| **NCAR**riers

| **NOR**Malized:POWer

| **OFF**Set

FREquency

| **DF**requency

| **[:CW]**

| **STAR**t

| **STOP**

| **F1[:CW]**

| **F2[:CW]**

| **FCEN**ter

| **[:CW]**

| **STAR**t

| **STOP**

| **CEN**Ter

| **SPAN**

HOPProduct?

IFBWidth

MAIN
IMTone
NORMalized
MODE
PMAP
INPut
OUTPut
TPOWER
COUPle[:STATe]
EQUalize:STATe
F1
F2
F1:START
F1:STOP
F2:START
F2:STOP
LEV
SET

Click on a [blue](#) keyword to view the command details.

Red commands are superseded.

Other Swept IMD commands

- [CALC:CUSTom:DEFine](#) - creates a Swept IMD measurement.
- [Swept IMD Calibration](#) - these are supplemental to the [Guided Cal](#) commands.
- Use std channel commands to set [Source](#) and [Receiver Attenuation](#).
- [SENS:ROLE:DEvice "RF2"](#) - use an external source for f2.

See Also

- **Example** [Create and Cal an IMD Measurement](#)

- [IM Spectrum commands](#)
 - [Learn about Swept IMD](#)
 - [Learn about Measurement Class](#)
 - [Synchronizing the PNA and Controller](#)
 - [SCPI Command Tree](#)
-

SENSe<num>:IMD:SWEep:TYPE <char>

(Read-Write) Sets and returns the sweep type for the IMD measurement.

Parameters

<num> Channel number of the IMD measurement. If unspecified, value is set to 1.

<char> Sweep type. Choose from:

- **FCENter** - (Center Frequency) Maintaining a constant tone spacing (DeltaF) and tone powers (P1 and P2), the center frequency (FC) is swept from Start to Stop.
- **DFrequency** - (Delta Frequency) The center frequency (FC) is held constant. The tone spacing is increased from StartDeltaF to StopDeltaF.
- **POWer** - The main tone frequencies are specified as either F1 and F2, or as FC and DeltaF. These frequencies are held constant while the power of each tone is stepped from the Start Power to the Stop Power.
- **CW** - The main tone frequencies (F1 and F2) and power levels (P1 and P2) are held constant. Measurements are taken for the specified Number of Points.
- **SEGment** - **Not available for IMDx.** Same as FCenter sweep, except that the center frequencies for the sweep are constructed using the standard [segment sweep commands](#).
- **LOPower** - All frequencies are fixed while the LO power is swept from Start to Stop power.

For each sweep type, use the commands that follow:

FCENter:

- [SENS:IMD:FREQ:FCEN:STAR](#)
- [SENS:IMD:FREQ:FCEN:STOP](#)
- [SENS:IMD:FREQ:FCEN:CEN](#)
- [SENS:IMD:FREQ:FCEN:SPAN](#)

- [SENS:IMD:FREQ:DFR:\[CW\]](#)
- [SENS:IMD:TPOW:F1](#)
- [SENS:IMD:TPOW:F2](#)

DFRequency

- [SENS:IMD:FREQ:DFR:STAR](#)
- [SENS:IMD:FREQ:DFR:STOP](#)
- [SENS:IMD:FREQ:FCEN:\[CW\]](#)
- [SENS:IMD:TPOW:F1](#)
- [SENS:IMD:TPOW:F2](#)

POWER

- [SENS:IMD:FREQ:F1:\[CW\]](#)
- [SENS:IMD:FREQ:F2:\[CW\]](#)
- [SENS:IMD:FREQ:FCEN:\[CW\]](#)
- [SENS:IMD:FREQ:DFR:\[CW\]](#)
- [SENS:IMD:TPOW:F1:STAR](#)
- [SENS:IMD:TPOW:F1:STOP](#)
- [SENS:IMD:TPOW:F2:STAR](#)
- [SENS:IMD:TPOW:F2:STOP](#)

CW

- [SENS:IMD:FREQ:F1:\[CW\]](#)
- [SENS:IMD:FREQ:F2:\[CW\]](#)
- [SENS:IMD:FREQ:FCEN:\[CW\]](#)
- [SENS:IMD:FREQ:DFR:\[CW\]](#)
- [SENS:IMD:TPOW:F1](#)
- [SENS:IMD:TPOW:F2](#)

SEGMENT - Not available for IMDx.

- [SENS:IMD:FREQ:DFR:\[CW\]](#)
- [SENS:IMD:TPOW:F1](#)
- [SENS:IMD:TPOW:F2](#)
- Use the standard [segment sweep commands](#) to set freq start and stop

LOPower - IMDx ONLY

- [SENS:MIX:LO:POW:STAR](#)
- [SENS:MIX:LO:POW:STOP](#)
- [SENS:IMD:FREQ:F1:\[CW\]](#)
- [SENS:IMD:FREQ:F2:\[CW\]](#)
- [SENS:IMD:FREQ:FCEN:\[CW\]](#)
- [SENS:IMD:FREQ:DFR:\[CW\]](#)
- [SENS:IMD:TPOW:F1](#)
- [SENS:IMD:TPOW:F2](#)

Examples

```
SENS:IMD:SWEep:TYPE CW
sense2:imd:sweep:type power
```

Query Syntax SENSE<cnum>:IMD:SWEep:TYPE?

Return Type Character

Default FCENTer

SENSe<cnum>:IMD:CSO:NDPProducts <num>

(Read-Write) Sets and returns the “N = number of distortion products” value for the calculation of the CSO parameter.
[Learn more.](#)

Parameters

- <cnum> Channel number of the IMD measurement. If unspecified, value is set to 1.
- <num> Number of distortion products.

Examples

```
SENS:IMD:CSO:NDPR 30  
sense2:imd:cso:ndproducts 7
```

Query Syntax SENSE<cnum>:IMD:CSO:NDPRoducts?

Return Type Numeric

Default 40

SENSE<cnum>:IMD:CSO:NORMalized:POWER <num>

(Read-Write) Sets and returns the CSO Power for POWER normalization mode. Valid only with measurement parameters: CSO2Lo and CSO2Hi and for Normalization Modes **dBm** and **dBmV**.
[Learn more.](#)

Parameters

- <cnum> Channel number of the IMD measurement. If unspecified, value is set to 1.
- <num> Power level. The units are determined by [Sens:IMD:Norm:Mode](#), which must be set first.

Examples

```
SENS:IMD:CSO:NORM:POW 0  
sense2:imd:cso:normalized:power -5
```

Query Syntax SENSE<cnum>:IMD:CSO:NORMalized:POWER?

Return Type Numeric

Default 0

SENSE<cnum>:IMD:CSO:OFFSet <num>

(Read-Write) Sets and returns the offset that is applied to CSO measurements. Valid only with measurement parameters: CSO2Lo and CSO2Hi. [Learn more.](#)

Parameters

- <cnum> Channel number of the IMD measurement. If unspecified, value is set to 1.
- <num> Offset value in dBm

Examples

```
SENS:IMD:CSO:OFFS 3  
sense2:imd:cso:offset 7
```

Query Syntax SENSE<cnum>:IMD:CSO:OFFSet?

Return Type Numeric

Default 0

SENSe<cnum>:IMD:CTB:NCARriers <num>

(Read-Write) Sets and returns the “N = Total number of carriers” value used in the calculation of the XMOD parameter. [Learn more.](#)

Parameters

- <cnum> Channel number of the IMD measurement. If unspecified, value is set to 1.
- <num> Number of carriers.

Examples

```
SENS:IMD:CTB:NCAR 10  
sense2:imd:ctb:ncarriers 50
```

Query Syntax SENSe<cnum>:IMD:CTB:NCARriers?

Return Type Numeric

Default 40

SENSe<cnum>:IMD:CTB:NORMalized:POWer <num>

(Read-Write) Sets and returns the CTB Power. Valid only with measurement parameters: CTBLo and CTBHi and for Normalization Modes **dBm** and **dBmV**. [Learn more.](#)

Parameters

- <num> Channel number of the IMD measurement. If unspecified, value is set to 1.
- <num> Power level. The units are determined by [Sens:IMD:Norm:Mode](#), which must be set first.

Examples

```
SENS:IMD:CTB:NORM:POW 0
sense2:imd:ctb:normalized:power -5
```

Query Syntax SENSE<num>:IMD:CTB:NORMALized:POWer?

Return Type Numeric

Default 0

SENSe<num>:IMD:CTB:OFFSet <num>

(Read-Write) Sets and returns the offset that is applied to CTB measurements. Valid only with measurement parameters: CTB, CTBLo, CTBHi, CTBE, CTBLo, and CTBEHi. [Learn more.](#)

Parameters

- <num> Channel number of the IMD measurement. If unspecified, value is set to 1.
- <num> Offset value in dBm

Examples

```
SENS:IMD:CTB:OFFS 3
sense2:imd:ctb:offset 7
```

Query Syntax SENSE<num>:IMD:CTB:OFFSet?

Return Type Numeric

Default 0

SENSe<num>:IMD:FREQuency:DFRequency[:CW] <num>

(Read-Write) Sets and returns fixed tone spacing.

Parameters

- <cnm> Channel number of the IMD measurement. If unspecified, value is set to 1.
- <num> Tone spacing frequency in Hz. The F1 and F2 tones MUST be within the frequency range of the PNA.

Examples

```
SENS:IMD:FREQ:DFR 1e6  
sense2:imd:frequency:dfrequency:cw 2e7
```

Query Syntax SENSE<cnm>:IMD:FREQuency:DFRequency[:CW]?

Return Type Numeric

Default 1 MHz

SENSe<cnm>:IMD:FREQuency:DFRequency:STARt <num>

(Read-Write) Sets and returns the starting main tone separation for sweep type = DFRequency (delta frequency).

Parameters

- <cnm> Channel number of the IMD measurement. If unspecified, value is set to 1.
- <num> Starting tone separation between F1 and F2 in Hz. Both F1 and F2 tones MUST be within the frequency range of the PNA where:

$$F1 \text{ (start)} = \text{FREQ:FCEN} - \text{DFR:Start} / 2$$

$$F2 \text{ (start)} = \text{FREQ:FCEN} + \text{DFR:Start} / 2$$

Examples

```
SENS:IMD:FREQ:DFR:STAR 1e6  
sense2:imd:frequency:dfrequency:start 2e7
```

Query Syntax SENSE<cnm>:IMD:FREQuency:DFRequency:STARt?

Return Type Numeric

Default 1 MHz

SENSe<cnm>:IMD:FREQuency:DFRequency:STOP <num>

(Read-Write) Sets and returns the stopping main tone separation for sweep type = DFREQUENCY (delta frequency).

Parameters

<num> Channel number of the IMD measurement. If unspecified, value is set to 1.

<num> Stopping tone separation between F1 and F2 in Hz. Both F1 and F2 tones MUST be within the frequency range of the PNA where:

$$F1 \text{ (stop)} = \text{FREQ:FCEN} - \text{DFR:Stop} / 2$$

$$F2 \text{ (stop)} = \text{FREQ:FCEN} + \text{DFR:Stop} / 2$$

Examples

```
SENS:IMD:FREQ:DFR:STOP 1e6
```

```
sense2:imd:frequency:dfrequency:stop 2e7
```

Query Syntax SENSE<num>:IMD:FREQUENCY:DFREQUENCY:STOP?

Return Type Numeric

Default 10 MHz

SENSe<num>:IMD:FREQUENCY:F1[:CW] <num>

(Read-Write) Sets and returns the frequency of the F1 tone.

Parameters

<num> Channel number of the IMD measurement. If unspecified, value is set to 1.

<num> F1 tone frequency in Hz. The F1 and F2 tones MUST be within the frequency range of the PNA.

Examples

```
SENS:IMD:FREQ:F1 1e9
```

```
sense2:imd:frequency:F1:cw 2e7
```

Query Syntax SENSe<num>:IMD:FREQUENCY:F1[:CW]?

Return Type Numeric

Default .9995 GHz

SENSe<num>:IMD:FREQUENCY:F2[:CW] <num>

(Read-Write) Sets and returns the frequency of the F2 tone.

Parameters

- <cnm> Channel number of the IMD measurement. If unspecified, value is set to 1.
- <num> F2 tone frequency in Hz. The F1 and F2 tones MUST be within the frequency range of the PNA.

Examples

```
SENS:IMD:FREQ:F2 1e9  
sense2:imd:frequency:F2:cw 2e7
```

Query Syntax SENSE<cnm>:IMD:FREQuency:F2[:CW]?

Return Type Numeric

Default 1.0005 GHz

SENSe<cnm>:IMD:FREQuency:FCENter[:CW] <num>

(Read-Write) Sets and returns the center frequency of the main tones.

Parameters

- <cnm> Channel number of the IMD measurement. If unspecified, value is set to 1.
- <num> Tone center frequency in Hz. The F1 and F2 tones MUST be within the frequency range of the PNA.

Examples

```
SENS:IMD:FREQ:FCEN 1e9  
sense2:imd:frequency:fcenter:cw 2e7
```

Query Syntax SENSE<cnm>:IMD:FREQuency:FCENter[:CW]?

Return Type Numeric

Default 1.0 GHz

SENSe<cnm>:IMD:FREQuency:FCENter:CENTer <num>

(Read-Write) Sets and returns the sweep center frequency when sweeping the main tones.

Parameters

- <cnm> Channel number of the IMD measurement. If unspecified, value is set to 1.
- <num> Center frequency in Hz. The F1 and F2 tones **MUST** be within the frequency range of the PNA.

Examples

```
SENS:IMD:FREQ:FCEN:CENT 1e9  
sense2:imd:frequency:fcenter:center 2e7
```

Query Syntax SENSE<cnm>:IMD:FREQuency:FCENter:CENTer?

Return Type Numeric

Default 13.255 GHz

SENSE<cnm>:IMD:FREQuency:FCENter:SPAN <num>

(Read-Write) Sets and returns the frequency span when sweeping the main tones.

Parameters

- <cnm> Channel number of the IMD measurement. If unspecified, value is set to 1.
- <num> Frequency span in Hz. The F1 and F2 tones **MUST** be within the frequency range of the PNA.

Examples

```
SENS:IMD:FREQ:FCEN:SPAN 1e9  
sense2:imd:frequency:fcenter:span 2e7
```

Query Syntax SENSE<cnm>:IMD:FREQuency:FCENter:SPAN?

Return Type Numeric

Default 26.489 GHz

SENSE<cnm>:IMD:FREQuency:FCENter:START <num>

(Read-Write) Sets and returns the start frequency when sweeping the main tones.

Parameters

- <cnm> Channel number of the IMD measurement. If unspecified, value is set to 1.
- <num> Start frequency in Hz. The F1 and F2 tones MUST be within the frequency range of the PNA.

Examples

```
SENS:IMD:FREQ:FCEN:STAR 1e9  
sense2:imd:frequency:fcenter:start 2e7
```

Query Syntax SENSE<cnm>:IMD:FREQuency:FCENter:STARt?

Return Type Numeric

Default 10.5 MHz

SENSE<cnm>:IMD:FREQuency:FCENter:STOP <num>

(Read-Write) Sets and returns the stop frequency when sweeping the main tones.

Parameters

- <cnm> Channel number of the IMD measurement. If unspecified, value is set to 1.
- <num> Stop frequency in Hz. The F1 and F2 tones MUST be within the frequency range of the PNA.

Examples

```
SENS:IMD:FREQ:FCEN:STOP 1e9  
sense2:imd:frequency:fcenter:stop 2e9
```

Query Syntax SENSE<cnm>:IMD:FREQuency:FCENter:STOP?

Return Type Numeric

Default 26.4995 GHz

SENSE:IMD:HOPROduct?

(Read-only) Returns the highest order product that can be measured by SweptIMD.

Parameters None

Examples

```
SENS:IMD:HOPR?  
'always returns 9
```

Return Type Numeric

Default 9

SENSe<num>:IMD:IFBWidth:MAIN <num>

(Read-Write) Sets and returns the IF Bandwidth for measurement of the main F1 and F2 tones. [Learn more about setting IFBW for IMD.](#)

Parameters

<num> Channel number of the IMD measurement. If unspecified, value is set to 1.

<num> Choose from: 1 | 2 | 3 | 5 | 7 | 10 | 15 | 20 | 30 | 50 | 70 | 100 | 150 | 200 | 300 | 500 | 700 | 1k | 1.5k | 2k | 3k | 5k | 7k | 10k | 15k | 20k | 30k | 50k | 70k | 100k | 150k | 200k | 280k | 360k | 600k

If an invalid number is specified, the analyzer will round up to the closest valid number.

Examples

```
SENS:IMD:IFBW:MAIN 280e3
```

```
sense2:imd:ifbwidth:main 150K
```

Query Syntax SENSe<num>:IMD:IFBWidth:MAIN?

Return Type Numeric

Default 1 kHz

SENSe<num>:IMD:IFBWidth:IMTone <num>

(Read-Write) Sets and returns the IF Bandwidth for measurement of the intermodulation products. [Learn more about setting IFBW for IMD.](#)

Parameters

<num> Channel number of the IMD measurement. If unspecified, value is set to 1.

<num> Choose from: 1 | 2 | 3 | 5 | 7 | 10 | 15 | 20 | 30 | 50 | 70 | 100 | 150 | 200 | 300 | 500 | 700 | 1k | 1.5k | 2k | 3k | 5k | 7k | 10k | 15k | 20k | 30k | 50k | 70k | 100k | 150k | 200k | 280k | 360k | 600k

If an invalid number is specified, the analyzer will round up to the closest valid number.

Examples

```
SENS:IMD:IFBW:IMT 50
```

```
sense2:imd:ifbwidth:imtone 200
```

Query Syntax SENSe<num>:IMD:IFBWidth:IMTone?

Return Type Numeric

Default 1 kHz

SENSe<cnum>:IMD:NORMalized:MODE <char>

(Read-Write) Sets and returns the method by which CTB and CSO calculations are performed.

Parameters

<cnum> Channel number of the IMD measurement. If unspecified, value is set to 1.

<char> Normalization mode. Choose from:

NONE - the normalized power is not used in calculation

NCARrier - CTB and CSO is corrected by subtracting $10 \cdot \log(N/2)$, where

- N = # of carriers for CTB
- N = # of distortion products for CSO

DBM - the composited normalized power for CTB or CSO is treated as a dBm value

DBmV - the composited normalized power for CTB or CSO is treated as a dBmV value.

Note: Power values are stored using the currently-set units. Therefore, first set units with this command, then set power values using:

[SENS:IMD:CSO:NORM:POW](#) or [SENS:IMD:CTB:NORM:POW](#)

Examples

```
SENS:IMD:NORM:MODE NCAR
```

```
sense2:imd:normalized:mode none
```

Query Syntax SENSe<cnum>:IMD:NORMalized:MODE?

Return Type Character

Default NCARrier

SENSe<cnum>:IMD:PMAP <input>,<output>

(Write-only) Sets the input port and output port of an IMD or IMDx channel. This setting is necessary only when using the **limited port mapping feature**. [Learn more.](#)

Parameters

- <num> Channel number of the IMD measurement. If unspecified, value is set to 1.
- <input> PNA port connected to the DUT Input. Choose from 1 or 3. When input is 3, an external combiner must be used.
- <output> PNA port connected to the DUT Output.

When input is 1, output must be 2.

When input is 3, output must be 4.

Examples

```
SENS:IMD:PMAP 3,4  
sense2:imd:pmap 3,4
```

Query Syntax Not Applicable

Default 1,2

SENSe<num>:IMD:PMAP:INPut?

(Read-only) Returns the PNA test port to be connected to the DUT input for an IMD or IMDx channel. Set the PNA port to DUT mapping using [SENS:IMD:PMAP](#)

Parameters

- <num> IMD channel number. If unspecified, value is set to 1.

Examples

```
SENS:IMD:PMAP:INP?  
sense2:imd:pmap:input?
```

Default 1

SENSe<num>:IMD:PMAP:OUTPut?

(Read-only) Returns the PNA test port to be connected to the DUT output for an IMD or IMDx channel. Set the PNA port to DUT mapping using [SENS:IMD:PMAP](#)

Parameters

<cnum> IMD channel number. If unspecified, value is set to 1.

Examples

```
SENS:IMD:PMAP:OUTP?  
sense2:imd:pmap:output?
```

Default 2

SENSe<cnum>:IMD:TPOWER:COUPlE[:STATe] <bool>

(Read-Write) Sets and returns the state of power coupling for F1 and F2. [Learn more about tone power.](#)

Parameters

<cnum> Channel number of the IMD measurement. If unspecified, value is set to 1.

<bool> Tone power level coupling state. Choose from:

ON (or 1) - F1 and F2 power is coupled.

OFF (or 0) - F1 and F2 power is NOT coupled. Set power levels individually.

Examples

```
SENS:IMD:TPOW:COUP 0  
sense2:imd:tpower:couple:state ON
```

Query Syntax SENSe<cnum>:IMD:TPOWER:COUPlE[:STATe]?

Return Type Boolean

Default ON

SENSe<cnum>:IMD:TPOWER:EQUalize:STATe <bool> - Superseded

(Read-Write) This command is replaced with [SENS:IMD:TPOW:LEV](#)

Sets and returns the state of Equal Tone Power setting. [Learn more about tone power.](#)

Parameters

<num> Channel number of the IMD measurement. If unspecified, value is set to 1.

<bool> Equal Tone Power state. Choose from:

ON (or 1) - Equalize f1 and f2 power.

OFF (or 0) - Do NOT equalize f1 and f2 power. Use source power cal.

Examples

```
SENS:IMD:TPOW:EQU 0  
sense2:imd:tpower:equalize:state ON
```

Query Syntax SENSE<num>:IMD:TPOWER:EQualize:STATe?

Return Type Boolean

Default OFF

SENSe<num>:IMD:TPOWER:F1 <num>

(Read-Write) Sets and returns the power level of the F1 tone. When [SENS:IMD:TPOW:COUP ON](#) (tone power is coupled), setting either F1 or F2 power sets both. [Learn more about tone power.](#)

Parameters

<num> Channel number of the IMD measurement. If unspecified, value is set to 1.

<num> Tone power level in dBm. Choose a value between +30 dBm and -30 dBm.

Examples

```
SENS:IMD:TPOW:F1 0  
sense2:imd:tpower:F1 -10
```

Query Syntax SENSE<num>:IMD:TPOWER:F1?

Return Type Numeric

Default -20 dBm

SENSe<num>:IMD:TPOWER:F2 <num>

(Read-Write) Sets and returns the power level of the F2 tone. When [SENS:IMD:TPOW:COUP ON](#) (tone power is coupled), setting either F1 or F2 power sets both. [Learn more about tone power.](#)

Parameters

- <num> Channel number of the IMD measurement. If unspecified, value is set to 1.
- <num> Tone power level in dBm. Choose a value between +30 dBm and -30 dBm.

Examples

```
SENS:IMD:TPOW:F2 0  
sense2:imd:tpower:F2 -10
```

Query Syntax SENSE<num>:IMD:TPOWER:F2?

Return Type Numeric

Default -20 dBm

SENSE<num>:IMD:TPOWER:F1:START <num>

(Read-Write) Sets and returns the start power level of the F1 tone. [Learn more about tone power.](#)

Parameters

- <num> Channel number of the IMD measurement. If unspecified, value is set to 1.
- <num> Start power in dBm. Choose a value between +30 dBm and -30 dBm.

Examples

```
SENS:IMD:TPOW:F1:STAR 0  
sense2:imd:tpower:F1:start -10
```

Query Syntax SENSE<num>:IMD:TPOWER:F1:START?

Return Type Numeric

Default -20 dBm

SENSE<num>:IMD:TPOWER:F1:STOP <num>

(Read-Write) Sets and returns the stop power level of the F1 tone. [Learn more about tone power.](#)

Parameters

<cnum> Channel number of the IMD measurement. If unspecified, value is set to 1.

<num> Stop power in dBm. Choose a value between +30 dBm and -30 dBm.

Examples

```
SENS:IMD:TPOW:F1:STOP 0  
sense2:imd:tpower:F1:stop 10
```

Query Syntax SENSE<cnum>:IMD:TPOWER:F1:STOP?

Return Type Numeric

Default -10 dBm

SENSE<cnum>:IMD:TPOWER:F2:START <num>

(Read-Write) Sets and returns the start power level of the F2 tone. [Learn more about tone power.](#)

Parameters

<cnum> Channel number of the IMD measurement. If unspecified, value is set to 1.

<num> Start power in dBm. Choose a value between +30 dBm and -30 dBm.

Examples

```
SENS:IMD:TPOW:F2:STAR 0  
sense2:imd:tpower:F2:start -10
```

Query Syntax SENSE<cnum>:IMD:TPOWER:F2:START?

Return Type Numeric

Default -20 dBm

SENSE<cnum>:IMD:TPOWER:F2:STOP <num>

(Read-Write) Sets and returns the stop power level of the F2 tone. [Learn more about tone power.](#)

Parameters

- <num> Channel number of the IMD measurement. If unspecified, value is set to 1.
<num> Stop power in dBm. Choose a value between +30 dBm and -30 dBm.

Examples

```
SENS:IMD:TPOW:F2:STOP 0  
sense2:imd:tpower:f2:stop 10
```

Query Syntax SENSE<num>:IMD:TPOWER:F2:STOP?

Return Type Numeric

Default -10 dBm

SENSe<num>:IMD:TPOWER:LEV <char>

(Read-Write) This command replaces SENS:IMD:TPOW:EQU and SENS:IMD:TPOW:SET

Sets and returns the tone power leveling mode.

Parameters

- <num> Channel number of the IMD measurement. If unspecified, value is set to 1.
<char> Choose from:

NONE - (Set Input Power) The specified f1 and f2 power levels are set at the DUT input.

INPUT - (Set Input Power, receiver leveling) The specified f1 and f2 power levels are set at the DUT input using receiver leveling at the input reference receiver.

EQUAL - (Set Input Power, equal tones at output) The specified f1 and f2 power levels are set at the DUT input and a measurement is made at the output.

OUTPUT - (Set Output Power, receiver leveling) The specified f1 and f2 power levels are set at the DUT output.

[Learn more about these choices.](#)

Examples

```
SENS:IMD:TPOW:LEV INPUT  
sense2:imd:tpower:lev output
```

Query Syntax SENSe<num>:IMD:TPOWER:LEV?

Return Type Character

Default NONE

SENSe<cnum>:IMD:TPOWER:SET <char> - Superseded

(Read-Write) This command is replaced with [SENS:IMD:TPOW:LEV](#)

Sets and returns whether tone power is specified at the DUT input or output.

Parameters

<cnum> Channel number of the IMD measurement. If unspecified, value is set to 1.

<char> Choose from:

INPUT - Specified power level is set at the DUT input.

OUTPUT - Specified power level is set at the DUT output.

Examples

```
SENS:IMD:TPOW:SET INPUT
```

```
sense2:imd:tpower:set output
```

Query Syntax SENSe<cnum>:IMD:TPOWER:SET?

Return Type Character

Default INPUT

Last Modified:

- 6-Nov-2012 Added power leveling mode (XC)
- Superseded two commands
- 18-Jun-2012 Added links to source and receiver attenuation
- 18-Apr-2012 Added link to Role:Device
- 5-May-2011 Added tone power settings
- 11-Aug-2009 Added PMAP
- 13-Mar-2009 Edited for IMDx
- 15-Aug-2008 MX New topic

Sense:IMS Commands

Controls IM Spectrum measurement configuration.

SENSe:IMS

PMAP

| **INPut**

| **OUTPut**

RBW

RESPonse

| **STARt**

| **STOP**

| **CENTer**

| **SPAN**

STIMulus

| **DFRequency**

| **FCENter**

| **F1FRequency**

| **F2FRequency**

| **TPOWer**

| **F1**

| **F2**

SWEep:

| **TYPE**

| **ORDer**

TPOWer

| **COUPle[:STATe]**

| **EQUal**

| **F1**

| [F2](#)

| [LEV](#)

| [SET](#)

TRACking

| [CHANnel](#)

| [MSENable](#)

| [SINDeX](#)

| [STATe](#)

Click on a [blue](#) keyword to view the command details.

See Also

- [About IM Spectrum measurements](#)
- [IMD Application](#)
- [Learn about Measurement Class](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

SENSe<num>:IMS:PMAP <input>,<output>

(Write-only) Sets the input port and output port of an IMS or IMSx channel. This setting is necessary only when using the **limited port mapping feature**. [Learn more.](#)

Parameters

<num> Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.

<input> PNA port connected to the DUT Input. Choose from 1 or 3. When input is 3, an external combiner must be used.

<output> PNA port connected to the DUT Output.

When input is 1, output must be 2.

When input is 3, output must be 4.

Examples `SENS:IMS:PMAP 3,4`
`sense2:ims:pmap 3,4`

Query Syntax Not Applicable

Default 1,2

SENSe<cnum>:IMS:PMAP:INPut?

(Read-only) Returns the PNA test port to be connected to the DUT input for an IMS or IMSx channel. Set the PNA port to DUT mapping using [SENS:IMS:PMAP](#)

Parameters

<cnum> IMS channel number. If unspecified, value is set to 1.

Examples `SENS:IMS:PMAP:INP?`
`sense2:ims:pmap:input?`

Default 1

SENSe<cnum>:IMS:PMAP:OUTPut?

(Read-only) Returns the PNA test port to be connected to the DUT output for an IMS or IMSx channel. Set the PNA port to DUT mapping using [SENS:IMS:PMAP](#)

Parameters

<cnum> IMS channel number. If unspecified, value is set to 1.

Examples `SENS:IMS:PMAP:OUTP?`
`sense2:ims:pmap:output?`

Default 2

SENSe<cnum>:IMS:RBW <num>

(Read-Write) Sets and returns the Resolution Bandwidth for the IM Spectrum measurement.

Parameters

<cnm> Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.

<num> Resolution BW in Hz. Choose from: 60k | 100k | 150k | 300k, 600k | 1.0M | 3.0M

If an invalid number is specified, the PNA will round up to the closest valid number.

Examples

```
SENS:IMS:RBW 600e3
```

```
sense2:ims:rbw 1MHz
```

Query Syntax SENSE<cnm>:IMS:RBW?

Return Type Numeric

Default 600 kHz

SENSe<cnm>:IMS:RESPonse:STARt <num>

(Read-Write) Sets and returns the receiver Start frequency for the IM Spectrum measurement. Valid ONLY when Tracking is NOT enabled and when Sweep Type = Linear. Otherwise, this setting is ignored.

Parameters

<cnm> Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.

<num> Start frequency in Hz. Choose a frequency within the range of the PNA.

Examples

```
SENS:IMS:RESP:STAR 1e9
```

```
sense2:ims:response:start 100e6
```

Query Syntax SENSe<cnm>:IMS:RESPonse:STARt?

Return Type Numeric

Default 950 MHz

SENSe<cnm>:IMS:RESPonse:STOP <num>

(Read-Write) Sets and returns the receiver Stop frequency for the IM Spectrum measurement. Valid ONLY when Tracking is NOT enabled and when Sweep Type = Linear. Otherwise, this setting is ignored.

Parameters

- <cnum> Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- <num> Stop frequency in Hz. Choose a frequency within the range of the PNA.

Examples

```
SENS:IMS:RESP:STOP 26e9  
sense2:ims:response:stop 100e6
```

Query Syntax SENSE<cnum>:IMS:RESPonse:STOP?

Return Type Numeric

Default 1.05 MHz

SENSe<cnum>:IMS:RESPonse:CENTer <num>

(Read-Write) Sets and returns the receiver Center frequency for the IM Spectrum measurement. Valid ONLY when Tracking is NOT enabled and when Sweep Type = Linear. Otherwise, this setting is ignored.

Parameters

- <cnum> Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- <num> Center frequency in Hz. Choose a frequency within the range of the PNA.

Examples

```
SENS:IMS:RESP:CENT 26e9  
sense2:ims:response:center 100e6
```

Query Syntax SENSe<cnum>:IMS:RESPonse:CENTer?

Return Type Numeric

Default 1.0 GHz

SENSe<cnum>:IMS:RESPonse:SPAN <num>

(Read-Write) Sets and returns the Span of receiver frequencies for the IM Spectrum measurement. Valid ONLY when Tracking is NOT enabled and when Sweep Type = Linear. Otherwise, this setting is ignored.

Parameters

- <cnm> Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- <num> Frequency span in Hz. All receiver frequencies should be within the range of the PNA.

Examples

```
SENS:IMS:RESP:SPAN 10e9  
sense2:ims:response:span 100e6
```

Query Syntax SENSE<cnm>:IMS:RESPonse:SPAN?

Return Type Numeric

Default 100 MHz

SENSe<cnm>:IMS:STIMulus:DFRequency <num>

(Read-Write) Sets and returns the DeltaF (tone spacing) for the IM Spectrum measurement. Valid ONLY when Tracking is NOT enabled. Otherwise, this setting is ignored.

Parameters

- <cnm> Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- <num> Delta frequency in Hz. All stimulus settings MUST be within the frequency range of the PNA.

Examples

```
SENS:IMS:STIM:DFR 1e6  
sense2:ims:stimulus:dfrequency 100e6
```

Query Syntax SENSE<cnm>:IMS:STIMulus:DFRequency?

Return Type Numeric

Default 10 MHz

SENSe<cnm>:IMS:STIMulus:FCENter <num>

(Read-Write) Sets and returns the Center frequency for the IM Spectrum measurement. Valid ONLY when Tracking is NOT enabled. Otherwise, this setting is ignored.

Parameters

- <cnm> Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- <num> Center frequency in Hz. All stimulus settings MUST be within the frequency range of the PNA.

Examples

```
SENS:IMS:STIM:FCEN 1e6  
sense2:ims:stimulus:fcenter 100e6
```

Query Syntax SENSE<cnm>:IMS:STIMulus:FCENter?

Return Type Numeric

Default 1.0 GHz

SENSe<cnm>:IMS:STIMulus:F1FRequency <num>

(Read-Write) Sets and returns the F1 frequency for the IM Spectrum measurement. Valid ONLY when Tracking is NOT enabled. Otherwise, this setting is ignored.

Parameters

- <cnm> Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- <num> F1 frequency in Hz. All stimulus settings MUST be within the frequency range of the PNA.

Examples

```
SENS:IMS:STIM:F1FR 1e6  
sense2:ims:stimulus:f1frequency 100e6
```

Query Syntax SENSe<cnm>:IMS:STIMulus:F1FRequency?

Return Type Numeric

Default 995 MHz

SENSe<cnm>:IMS:STIMulus:F2FRequency <num>

(Read-Write) Sets and returns the F2 frequency for the IM Spectrum measurement. Valid ONLY when Tracking is NOT enabled. Otherwise, this setting is ignored.

Parameters

- <num> Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- <num> F2 frequency in Hz. All stimulus settings MUST be within the frequency range of the PNA.

Examples

```
SENS:IMS:STIM:F2FR 1e6  
sense2:ims:stimulus:f2frequency 100e6
```

Query Syntax SENSE<num>:IMS:STIMulus:F2FRequency?

Return Type Numeric

Default 1.005 MHz

SENSe<num>:IMS:SWEep:TYPE <char>

(Read-Write) Sets and returns the method in which the spectrum of signals to view are specified. When Tracking is enabled the frequency of the main tones (f1 and f2) are always determined by the Swept IMD channel.

Parameters

- <num> Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- <char> IM Spectrum sweep type. Choose from:
- **LINear** When Tracking is enabled, allows tuning the Response Settings (receiver) to any values within the frequency range of the PNA. When Tracking is NOT enabled also allows setting the Stimulus (sources) to any values within the frequency range or the PNA.
 - **SECond** The receiver is tuned to view the 2nd order products (f2-f1 and f1+f2) of the main tones that are currently specified in Stimulus Settings. When Tracking is enabled, the main tones are specified in the Swept IMD channel.
 - **THIRd** The receiver is tuned to view the 3rd order products (2f1 -f2 and 2f2-f1) of the main tones that are currently specified in Stimulus Settings. When Tracking is enabled, the main tones are specified in the Swept IMD channel.
 - **NTH** The frequency range is set to N * DeltaF. This algorithm will NOT tune the receivers to view the EVEN order products.

Examples `SENS:IMS:SWEep:TYPE LIN`

`sense2:ims:sweep:type nth`

Query Syntax `SENSe<cnum>:IMS:SWEep:TYPE?`

Return Type Character

Default NTH

SENSe<cnum>:IMS:SWEep:ORDer <num>

(Read-Write) Sets and returns the order number of signals to view when [SENS:IMS:SWE:TYPE NTH](#) is specified.

Parameters

<cnum> Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.

<num> Order number of IM products to view. The frequency range is set to N (this number) x DeltaF (set with [SENS:IMS:STIM:DFR](#)).

Examples `SENS:IMS:SWEep:ORD 5`

`sense2:ims:sweep:order 12`

Query Syntax `SENSe<cnum>:IMS:SWEep:ORDer?`

Return Type Numeric

Default 9

SENSe<cnum>:IMS:TPOWER:COUPl[e[:STATe]] <bool>

(Read-Write) Sets and returns the state of power coupling for F1 and F2.

Parameters

- <cnum> Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- <bool> Tone power level coupling. Choose from:
 - ON (or 1)** - F1 and F2 power is coupled.
 - OFF (or 0)** - F1 and F2 power is NOT coupled. Set power levels individually.

Examples

```
SENS:IMS:TPOW:COUP 0  
sense2:ims:tpower:couple:state ON
```

Query Syntax SENSE<cnum>:IMS:TPOWER:COUPLE[:STATE]?

Return Type Boolean

Default ON

SENSe<cnum>:IMS:TPOWER:EQUalize:STATE <bool> - Superseded

(Read-Write) This command is replaced with [SENS:IMS:TPOW:LEV](#)

Sets and returns the state of Equal Tone Power setting. [Learn more about tone power.](#)

Parameters

- <cnum> Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- <bool> Equal Tone Power state. Choose from:
 - ON (or 1)** - Equalize f1 and f2 power.
 - OFF (or 0)** - Do NOT equalize f1 and f2 power. Use source power cal.

Examples

```
SENS:IMS:TPOW:EQU 0  
sense2:ims:tpower:equalize:state ON
```

Query Syntax SENSE<cnum>:IMS:TPOWER:EQUalize:STATE?

Return Type Boolean

Default OFF

SENSe<cnum>:IMS:STIMulus:TPOWER:F1 <value>

(Read-Write) Sets and returns the power level of the F1 tone. When [SENS:IMS:TPOW:COUP ON](#) (tone power is coupled), setting either F1 or F2 power sets both. This setting is ignored if [SENS:IMS:TRAC:STAT](#) is enabled.

Parameters

- <cnum> Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- <value> F1 tone power level in dBm. Choose a value between +30 dBm and -30 dBm.

Examples

```
SENS:IMS:STIM:TPOW:F1 -10  
sense2:ims:stimulus:tpower:f1 0
```

Query Syntax SENSE<cnum>:IMS:STIMulus:TPOWer:F1?

Return Type Numeric

Default -20

SENSe<cnum>:IMS:STIMulus:TPOWer:F2 <value>

(Read-Write) Sets and returns the power level of the F2 tone. When [SENS:IMS:TPOW:COUP ON](#) (tone power is coupled), setting either F1 or F2 power sets both. This setting is ignored if [SENS:IMS:TRAC:STAT](#) is enabled.

Parameters

- <cnum> Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.
- <value> F2 tone power level in dBm. Choose a value between +30 dBm and -30 dBm.

Examples

```
SENS:IMS:STIM:TPOW:F2 -10  
sense2:ims:stimulus:tpower:f2 0
```

Query Syntax SENSe<cnum>:IMS:STIMulus:TPOWer:F2?

Return Type Numeric

Default -20

SENSe<cnum>:IMS:TPOWer:LEV <char>

(Read-Write) This command replaces SENS:IMS:TPOW:EQU and SENS:IMS:TPOW:SET

Sets and returns the tone power leveling mode.

Parameters

<cnun> Channel number of the IMD measurement. If unspecified, value is set to 1.

<char> Choose from:

NONE - (Set Input Power) The specified f1 and f2 power levels are set at the DUT input.

INPUT - (Set Input Power, receiver leveling) The specified f1 and f2 power levels are set at the DUT input using receiver leveling at the input reference receiver.

EQUAL - (Set Input Power, equal tones at output) The specified f1 and f2 power levels are set at the DUT input and a measurement is made at the output.

OUTPUT - (Set Output Power, receiver leveling) The specified f1 and f2 power levels are set at the DUT output.

[Learn more about these choices.](#)

Examples

```
SENS:IMS:TPOW:LEV INPUT
```

```
sense2:ims:tpower:lev output
```

Query Syntax SENSE<cnun>:IMS:TPOWER:LEV?

Return Type Character

Default NONE

SENSe<cnun>:IMS:TPOWER:SET <char> - Superseded

(Read-Write) This command is replaced with [SENS:IMS:TPOW:LEV](#)

Sets and returns whether tone power is specified at the DUT input or output.

Parameters

<num> Channel number of the IMSpectrum measurement. If unspecified, value is set to 1.

<char> Choose from:

INPUT - Specified power level is set at the DUT input.

OUTPUT - Specified power level is set at the DUT output.

Examples

```
SENS:IMS:TPOW:SET INPUT
```

```
sense2:ims:tpower:set output
```

Query Syntax SENSE<num>:IMS:TPOWER:SET?

Return Type Character

Default INPUT

SENSe<num>:IMS:TRACking:CHANnel <num>

(Read-Write) Sets and returns the IMD channel number to which the IM Spectrum channel is coupled.

Parameters

<num> IMS channel. If unspecified, value is set to 1.

<num> Existing IMD channel to which frequency and power settings are coupled.

Examples

```
SENS:IMS:TRAC:CHAN 2
```

```
sense2:ims:tracking:channel 1
```

Query Syntax SENSe<num>:IMS:TRACking:CHANnel?

Return Type Numeric

Default First IMD channel

SENSe<num>:IMS:TRACking:MSEnable <bool>

(Read-Write) Sets and returns the step sweep mode for the IM Spectrum channel.

Parameters

<num> IMS channel. If unspecified, value is set to 1.

<bool> Choose from:

OFF (or 0) - Automatic Step

ON (or 1) - Manual Step

Examples

```
SENS:IMS:TRAC:MSEN 1
```

```
sense2:ims:tracking:msenable 0
```

Query Syntax SENSE<num>:IMS:TRACking:MSENable?

Return Type Boolean

Default 0 - Automatic

SENSE<num>:IMS:TRACking:SINDEX <num>

(Read-Write) When [SENS:IMS:TRAC:MSEN = Manual](#), sets and returns the data point number at which a sweep is performed.

Parameters

<num> IMS channel. If unspecified, value is set to 1.

<num> Step index. Choose from 1 to the specified number of points.

Examples

```
SENS:IMS:TRAC:SINDEX 201
```

```
sense2:ims:tracking:sindex 1
```

Query Syntax SENSE<num>:IMS:TRACking:SINDEX?

Return Type Numeric

Default 1

SENSE<num>:IMS:TRACking:STATE <bool>

(Read-Write) When an IMD channel exists, allows the IM Spectrum frequency and power setting to track (couple with) the IMD channel settings.

Parameters

<num> IMS channel. If unspecified, value is set to 1.

<bool> Tracking state. Choose from:

ON (or 1) - IM Spectrum frequency and power settings track the IMD channel settings.

OFF (or 0) - IM Spectrum frequency and power settings are specified in the IMS channel.

Examples

```
SENS:IMS:TRAC:STAT 0  
sense2:ims:tracking:state ON
```

Query Syntax SENSE<num>:IMS:TRACking:STATe?

Return Type Boolean

Default OFF

Last Modified:

- 14-Nov-2012 Added Leveling command
- Superseded two commands
- 5-May-2011 Added tone power settings
- 11-Aug-2009 Added PMAP (9.0)
- 12-Mar-2014 MX New topic

Sense:Mixer Commands

Performs Mixer setup and configuration.

SENSe:MIXer:

APPLy

AVOidspurs

CALC

DISCard

ELO - More Commands

IF:FREQ:

| **SIDeband**

| **STARt**

| **STOP**

INPut:FREQ:

| **DENominator**

| **FIXed**

| **MODE**

| **NUMerator**

| **STARt**

| **STOP**

INPut:POWer

| **USENominal**

LO:FREQ:

| **DENominator**

| **FIXed**

| **ILTI**

| **MODE**

| **NUMerator**

| **STARt**

| **STOP**

LO:NAME

LO:POWer

| **STARt**

| **STOP**

LOAD

NORMalize:POINT

OUTPut:FREQ:

| **FIXed**

| **MODE**

| **SIDeband**

| **STARt**

| **STOP**

[PHASe](#)
[PMAP](#)
 | [INPut](#)
 | [OUTPut](#)
[RECalculate](#)
[REVerse](#)
[ROLE](#)
 | [CATalog?](#)
 | [DEVice](#)
[SAVE](#)
[SEGMENT](#) More Commands
[STAGE](#) (number of LOs)
[XAXis](#)

Click on a [blue](#) keyword to view the command details.

See Also

- [Example Programs](#)
- [Learn about the Frequency Converter Application](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

Note: If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

SENSe<ch>:MIXer:APPLy

(Write only) Applies the mixer setup settings and turns the channel ON. (Performs the same function as the Apply button on the [mixer setup dialog box](#)).

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

Examples `SENS:MIX:APPL`

Query Syntax Not Applicable

Default Not Applicable

SENSe<ch>:MIXer:AVOIdspurs <bool>

(Read Write) Sets and returns the state of the avoid spurs feature. [Learn more about avoid spurs.](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <bool> Avoid spurs state. Choose from
 - 0 - Avoid spurs OFF
 - 1 - Avoid spurs ON

Examples

```
SENS:MIX:AVO  
sense2:mixer:avoidspurs 1
```

Query Syntax SENSE<ch>:MIXer:AVOidspurs?

Return Type Boolean

Default 0 (OFF)

SENSe<ch>:MIXer:CALCulate <char>

(Write only) Calculates the Input, IF, or Output frequencies of the mixer setup and updates the channel settings.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <char> Mixer port to be calculated. Choose from:

<char>	1st or only stage requires:	In addition, 2nd stage requires:
INPut	<ul style="list-style-type: none">• Output Start/Stop/Fixed frequencies• LO Start/Stop/Fixed frequencies• Output sideband (High or Low)	<ul style="list-style-type: none">• IF Start/Stop/Fixed frequencies• 2nd Start/Stop/Fixed frequencies• IF sideband (High or Low)
BOTH	NA	<ul style="list-style-type: none">• IF Start/Stop/Fixed frequencies• Both Start/Stop/Fixed frequencies
OUTPut	<ul style="list-style-type: none">• Input Start/Stop/Fixed	<ul style="list-style-type: none">• IF Start/Stop/Fixed

	<p>frequencies</p> <ul style="list-style-type: none"> • LO Start/Stop/Fixed frequencies • Output sideband (High or Low) 	<p>frequencies</p> <ul style="list-style-type: none"> • 2nd Start/Stop/Fixed frequencies • IF sideband (High or Low)
LO_1	<ul style="list-style-type: none"> • Input Start/Stop/Fixed frequencies • Output Start/Stop/Fixed frequencies • Output sideband (High or Low) 	<ul style="list-style-type: none"> • IF Start/Stop/Fixed frequencies • 2nd Start/Stop/Fixed frequencies • IF sideband (High or Low)
LO_2	NA	<ul style="list-style-type: none"> • Input Start/Stop/Fixed frequencies • 1st LO Start/Stop/Fixed frequencies • Output Start/Stop/Fixed frequencies • IF sideband(High or Low) • Output sideband(High or Low)

Examples

SENS:MIX:CALC Output

Query Syntax Not Applicable

Default Not Applicable

SENSe<ch>:MIXer:DISCard

(Write only) Cancels changes that have been made to the Converter setup and reverts to the previously-saved setup. Same as the Cancel button on the [mixer setup dialog box](#).

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

Examples

SENS:MIX:DISC

Query Syntax Not Applicable

Default Not Applicable

SENSe<ch>:MIXer:IF:FREQuency:SIDeband <char>

(Read-Write) When [two LO stages are used](#), sets or returns whether to select the sum or difference for the IF1 product. (Input + or - LO1 = IF1)



- This setting corresponds to the   buttons on LO1 on the [Mixer setup dialog](#)
- This setting is ignored when [ONE LO stage](#) is selected.
- Also set [SENS:MIX:OUTP:FREQ:SID](#) to LOW or HIGH to determine the output frequency of the mixer.

[See Note](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
- <char> Sideband value. Choose from
- LOW** - Difference (-)
- HIGH** - Sum (+)

Examples

```
SENS:MIX:IF:FREQ:SID LOW  
SENSe2:MIXer:IF:FREQ:SIDeband HIGH
```

Query Syntax SENSe<ch>:MIXer:IF:FREQuency:SIDeband?

Return Type Character

Default LOW

SENSe<ch>:MIXer:IF:FREQuency:STARt <num>

(Read-Write) Sets or returns the IF start frequency value of the mixer. [See Note](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
- <num> IF Start Frequency value

Examples

```
SENS:MIX:IF:FREQ:STAR 1e9  
SENSe2:MIXer:IF:FREQ:STARt 1000000000
```

Query Syntax SENSe<ch>:MIXer:IF:FREQuency:STARt?

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:IF:FREQuency:STOP <num>

(Read-Write) Sets or returns the stop frequency value of the mixer IF frequency. [See Note](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
<num> IF Stop Frequency value

Examples

```
SENS:MIX:IF:FREQ:STOP 2e9  
SENSe2:MIXer:IF:FREQ:STOP 2000000000
```

Query Syntax SENSe<ch>:MIXer:IF:FREQuency:STOP?

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:INPut:FREQuency:DENominator <value>

(Read-Write) Sets or returns the denominator value of the Input Fractional Multiplier. [See Note](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
<value> Input denominator value.

Examples

```
SENS:MIX:INP:FREQ:DEN 5  
SENSe2:MIXer:INPut:FREQ:DENominator 4
```

Query Syntax SENSe<ch>:MIXer:INPut:FREQuency:DENominator?

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:INPut:FREQuency:FIXed<value>

(Read-Write) Sets or returns the fixed frequency of the input. [See Note](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
- <value> Input frequency.

Examples

```
SENSe:MIXer:INPut:FREQ:FIXed 1e9  
SENSe2:MIXer:INPut:FREQ:FIXed 1000000000
```

Query Syntax SENSE<ch>:MIXer:INPut:FREQuency:FIXed?

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:INPut:FREQuency:MODE <char>

(Read-Write) Sets or returns the Input sweep mode.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
- <char> Input sweep mode. Choose either **FIXED** or **SWEPT**

Examples

```
SENS:MIX:INP:FREQ:MODE FIXED  
SENSe2:MIXer:INP:FREQ:MODE swept
```

Query Syntax SENSE<ch>:MIXer:INPut:FREQuency:MODE?

Return Type Character

Default Fixed

SENSe<ch>:MIXer:INPut:FREQuency:NUMerator <value>

(Read-Write) Sets or returns the numerator value of the Input Fractional Multiplier. [See Note](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
- <value> Input numerator value.

Examples

```
SENS:MIX:INP:FREQ:NUM 3  
SENSe2:MIXer:INPut:FREQ:NUMerator 1
```

Query Syntax SENSE<ch>:MIXer:INPut:FREQ:NUMerator?

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:INPut:FREQuency:STARt <value>

(Read-Write) Sets or returns the Input start frequency value of the mixer. [See Note](#)

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<value> Input Start frequency

Examples

```
SENS:MIX:INP:FREQ:STAR 1e9
SENSe2:MIXer:INPut:FREQ:STARt 1000000000
```

Query Syntax SENSe<ch>:MIXer:INPut:FREQuency:STARt?

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:INPut:FREQuency:STOP <value>

(Read-Write) Sets or returns the Input stop frequency value of the mixer. [See Note](#)

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<value> Input stop frequency

Examples

```
SENS:MIX:INP:FREQ:STOP 2e9
SENSe2:MIXer:INPut:FREQ:STOP 2000000000
```

Query Syntax SENSe<ch>:MIXer:INPut:FREQuency:STOP?

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:INPut:POWer <value>

(Read-Write) Sets or returns the value of the Input Power.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
- <value> Input power in dBm.

Examples

```
SENS:MIX:INP:POW 9  
SENSe2:MIXer:INPut:POWer 5
```

Query Syntax SENSE<ch>:MIXer:INPut:POWer?

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:INPut:POWer:USENominal <bool>

(Read-Write) Toggles the Use Nominal Incident Power setting ON and OFF. This setting is ONLY to be used with SMC measurements. [Learn more about Nominal Incident Power.](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
- <value> (**boolean**) - Nominal Incident Power State. Choose from:
 - ON (1) - Turn nominal incident power ON
 - OFF (0) - Turn nominal incident power OFF

Examples

```
SENS:MIX:INP:POW:USEN 1  
SENSe2:MIXer:INPut:POWer:USENominal OFF
```

Query Syntax SENSE<ch>:MIXer:INPut:POWer:USENominal?

Return Type Boolean

Default OFF

SENSe<ch>:MIXer:LO<n>:FREQUENCY:DENominator <value>

(Read-Write) Sets or returns the denominator value of the LO Fractional Multiplier. [See Note](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
- <n> LO stage number. Choose 1 or 2.
- <value> LO denominator.

Examples

```
SENS:MIX:LO:FREQ:DEN 5  
SENSe2:MIXer:LO2:FREQ:DENominator 4
```

Query Syntax SENSE<ch>:MIXer:LO<n>:FREQUency:DENominator?

Return Type Numeric

Default 1

SENSe<ch>:MIXer:LO<n>:FREQUency:FIXed <value>

(Read-Write) Sets or returns the fixed frequency of the specified mixer LO. [See Note](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
- <n> LO stage number. Choose 1 or 2
- <value> LO frequency.

Examples

```
SENS:MIX:LO:FREQ:FIX 1e9  
SENSe2:MIXer:LO2:FREQ:FIXed 1000000000
```

Query Syntax SENSE<ch>:MIXer:LO<n>:FREQUency:FIXed?

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:LO<n>:FREQUency:ILTI <bool>

(Read-Write) Specifies whether to use the Input frequency that is **greater than** the LO or **less than** the LO. To learn more, see the [mixer setup](#) dialog box help.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
- <n> LO stage number. Choose **1** or **2**
- <bool> **ON (1)** - Use the Input that is Greater than the specified LO.
OFF (0) - Use the Input that is Less than the specified LO.

Examples

```
SENS:MIX:LO1:FREQ:ILTI 1  
sense2:mixer:lo2:frequency:ilti ON
```

Query Syntax SENSE<ch>:MIXer:LO<n>:FREQuency:ILTI?

Return Type Boolean

Default OFF

SENSe<ch>:MIXer:LO<n>:FREQuency:MODE <char>

(Read-Write) Sets or returns the LO sweep mode.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
- <n> LO stage number. Choose **1** or **2**
- <char> LO sweep mode. Choose either **FIXED** or **SWEPT**

Examples

```
SENS:MIX:LO:FREQ:MODE FIXED  
SENSe2:MIXer:LO2:FREQ:MODE swept
```

Query Syntax SENSe<ch>:MIXer:LO<n>:FREQuency:MODE?

Return Type Character

Default Fixed

SENSe<ch>:MIXer:LO<n>:FREQuency:NUMerator <value>

(Read-Write) Sets or returns the numerator value of the LO Fractional Multiplier. [See Note](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
- <n> LO stage number. Choose **1** or **2**
- <value> LO Numerator.

Examples

```
SENS:MIX:LO:FREQ:NUM 5  
SENSe2:MIXer:LO2:FREQ:NUMerator 4
```

Query Syntax SENSE<ch>:MIXer:LO<n>:FREQuency:NUMerator?

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:LO<n>:FREQuency:STARt <value>

(Read-Write) Sets or returns the LO start frequency value. [See Note](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
- <n> LO stage number. Choose **1** or **2**
- <value> LO Start Frequency in Hertz.

Examples

```
SENS:MIX:LO:FREQ:STAR 5E9
```

Query Syntax SENSE<ch>:MIXer:LO<n>:FREQuency:STARt?

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:LO<n>:FREQuency:STOP <value>

(Read-Write) Sets or returns the LO stop frequency value. [See Note](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
- <n> LO stage number. Choose 1 or 2
- <value> LO Stop Frequency in Hertz.

Examples

```
SENS:MIX:LO:FREQ:STOP 5E9
```

Query Syntax SENSE<ch>:MIXer:LO<n>:FREQuency:STOP?

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:LO<n>:NAME <value>

(Read-Write) Sets or returns the name of the PNA internal source or external source to use as the LO in a converter measurement.

Important Note: This setting is immediately send to the channel configuration. First set and apply mixer frequency settings, then send this command. Otherwise, 'invalid setting' errors may occur.

See [Remotely Specifying a Source Port](#).

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
- <n> LO stage number. Choose 1 or 2.
- <value> (**string**) - LO Source name. Use [Source:CAT?](#) to return a list of valid source ports. An external source must be configured and selected to be valid. [Learn more about external source configuration.](#)

Examples

```
SENS:MIX:LO:NAME "MySource"
```

Query Syntax SENSe<ch>:MIXer:LO<n>:NAME?

Return Type String

Default "Not Controlled"

SENSe<ch>:MIXer:LO<n>:POWER <value>

(Read-Write) Sets or returns the LO Power fixed value.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
- <n> LO stage. Choose **1** or **2**
- <value> LO Power in dBm

Examples `SENS:MIX:LO:POW 9`

Query Syntax `SENSe<ch>:MIXer:LO<n>:POWER?`

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:LO<n>:POWER:STARt <value>

(Read-Write) For an LO power sweep, sets or returns the LO power start value.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
- <n> LO stage. Choose **1**
- <value> LO start power in dBm

Examples `SENS:MIX:LO1:POW:STAR -10`

Query Syntax `SENSe<ch>:MIXer:LO1:POWER:STARt?`

Return Type Numeric

Default - 20 dBm

SENSe<ch>:MIXer:LO<n>:POWER:STOP <value>

(Read-Write) For an LO power sweep, sets or returns the LO power stop value.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
- <n> LO stage. Choose 1
- <value> LO stop power in dBm

Examples `SENS:MIX:LO1:POW:STOP 10`

Query Syntax `SENSe<ch>:MIXer:LO1:POWer:STOP?`

Return Type Numeric

Default -10 dBm

SENSe<ch>:MIXer:LOAD <name>

(Write-only) Loads a previously-configured mixer attributes file (.mxr)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
- <name> Path and file name (including .mxr extension) to load.

Examples `SENSe:MIXer:LOAD "C:/Program Files/Agilent/Network Analyzer/Documents/Mixer/MyMixer.mxr"`

Default Not Applicable

SENSe<ch>:MIXer:NORMalize:POINT <value>

(Read-Write) Sets or returns the data point for normalizing the phase measurement. [Learn more.](#)

Parameters

- <ch> Channel number of the SMC measurement. If unspecified, value is set to 1.
- <value> Normalization data point. Choose a data point number between 1 and the max number of data points in the sweep that has the least amount of expected noise.

Examples `SENS:MIX:NORM:POIN 101`
`sense2:mixer:normalize:point 50`

Query Syntax `SENSe<ch>:MIXer:NORMalize:POINT?`

Return Type Numeric

Default Middle point in the sweep

SENSe<ch>:MIXer:OUTPut:FREQuency:FIXed <value>

(Read-Write) Sets or returns the output fixed frequency of the mixer. [See Note](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
<value> Output fixed frequency in Hertz.

Examples

```
SENS:MIX:OUTP:FREQ:FIX 5e9
```

Query Syntax SENSe<ch>:MIXer:OUTPut:FREQuency:FIXed?

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:OUTPut:FREQuency:MODE <char>

(Read-Write) Sets or returns the Output sweep mode.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
<char> Output sweep mode. Choose either **FIXED** or **SWEPT**

Examples

```
SENS:MIX:OUT:FREQ:MODE FIXED  
SENSe2:MIXer:OUTput:FREQuency:MODE swept
```

Query Syntax SENSe<ch>:MIXer:OUTPut:FREQuency:MODE?

Return Type Character

Default Fixed

SENSe<ch>:MIXer:OUTPut:FREQuency:SIDeband <value>

(Read-Write) Specify whether to select the sum (High) or difference (Low) products.

- When one LO is used: Input + or - LO1 = Output frequency
- When two LOs are used: IF1 + or - LO2 = Output frequency

Use [SENS:MIX:IF:FREQ:SID](#) when two LOs are used to determine the IF1 frequency.

Use [Sens:Mixer:Stage](#) to set 1 or 2 LOs

[See Note](#)

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<value> Sideband value. Choose from

LOW - Low or Difference (-)

HIGH - High or Sum (+)

Examples

```
SENS:MIX:OUTP:FREQ:SID LOW
SENSe2:MIXer:OUTPut:FREQ:SIDeband HIGH
```

Query Syntax SENSE<ch>:MIXer:OUTPut:FREQuency:SIDeband?

Return Type Character

Default LOW

SENSe<ch>:MIXer:OUTPut:FREQuency:START <value>

(Read-Write) Sets or returns the Output start frequency of the mixer. [See Note](#)

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<value> Output start frequency

Examples

```
SENS:MIX:OUTP:FREQ:STAR 1e9
SENSe2:MIXer:OUTPut:FREQ:START 1000000000
```

Query Syntax SENSE<ch>:MIXer:OUTPut:FREQuency:START?

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:OUTPut:FREQuency:STOP <value>

(Read-Write) Sets or returns the Output stop frequency of the mixer. [See Note](#)

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
- <value> Output stop frequency

Examples

```
SENS:MIX:OUTP:FREQ:STOP 1e9  
SENSe2:MIXer:OUTPut:FREQ:STOP 1000000000
```

Query Syntax SENSE<ch>:MIXer:OUTPut:FREQUency:STOP?

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:PHASe <bool>

(Read Write) Sets and returns the state of SMC Phase measurements and calibrations. [Learn more.](#)

In the User Interface, there are two "enable phase" checkboxes: in the [Phase Settings dialog](#) and in the [Calibration Wizard](#). Checking one enables both. This single command also enables both.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <bool> Include Phase measurement state. Choose from
 - ON or 1 - Include phase in SMC measurements
 - OFF or 0 - Do NOT include phase in SMC measurements

Examples

```
SENS:MIX:PHAS 1  
sense2:mixer:phase off
```

Query Syntax SENSE<ch>:MIXer:PHASe?

Return Type Boolean

Default 0 (OFF)

SENSe<ch>:MIXer:PMAP <in>,<out>

(Write-only) Sets the PNA to DUT port map for FCA measurements. Use SENS:MIX:PMAP:INP? and SENS:MIX:PMAP:OUTP? to read these values. Learn about [selectable FCA DUT ports](#).

Changing the ports may limit your ability to use an internal second source. If a selected port is shared by one of the sources, then that source will not be available as an LO source. [Learn more about Internal second sources](#).

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.
- <in> PNA port to connect to the DUT input.
- For SMC, choose any unused PNA port.
 - For VMC, set to 1.
- <out> PNA port to connect to the DUT output. Choose any unused port for SMC and VMC.

Examples

```
SENS:MIX:PMAP 2,1  
sense2:mixer:pmap 4,2
```

Query Syntax Not Applicable

Default 1,2

SENSe<ch>:MIXer:PMAP:INPut?

(Read-only) Returns the PNA port that is mapped to the DUT input. Use [SENS:MIX:PMAP](#) to set this value.

Learn about [selectable FCA DUT ports](#).

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1.

Examples

```
SENS:MIX:PMAP:INP?  
sense2:mixer:pmap:input?
```

Default 1

SENSe<ch>:MIXer:PMAP:OUTPut?

(Read-only) Returns the PNA port that is mapped to the DUT output. Use [SENS:MIX:PMAP](#) to set this value.

Learn about [selectable FCA DUT ports](#).

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

Examples `SENS:MIX:PMAP:OUTP?`
`sense2:mixer:pmap:output?`

Default 2

SENSe<ch>:MIXer:RECalculate

(Write only) Repeats the last calculation that was performed, including all ON (state) segments in segment table.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

Examples `SENS:MIX:REC`

Query Syntax Not Applicable

Default Not Applicable

SENSe<ch>:MIXer:REVerse <bool>

(Read-Write) Sets whether to include SC12 sweeps during measurements.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<bool> (Boolean) Choose from:

ON (1) - Include the SC12 (reverse) sweep.

OFF (0) - Do NOT Include the SC12 (reverse) sweep.

Examples `SENS:MIX:REV 1`
`sense2:mixer:reverse ON`

Query Syntax SENSe<ch>:MIXer:REVerse?

Return Type Boolean

Default ON (1)

SENSe<ch>:MIXer:ROLE:CATalog? - Superseded

(Read-only) This command is replaced with [SENSe:ROLE:CATalog](#) which can be used by all channels.

Returns a list of valid roles for the IMD Converter application.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

Examples

```
SENS:MIX:ROLE:CAT?  
sense2:mixer:role:catalog?
```

Default Not Applicable

SENSe<ch>:MIXer:ROLE:DEvice <role>,<source> Superseded

(Read-Write) This command is replaced with [SENSe:ROLE:DEvice](#) which can be used by all channels.

Assigns a configured external source to the specified role for the converter application.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<role> (String) Role to which the external source is assigned. Choose from:

For IMDX and IMSX, choose from:

"RF2"

"LO1"

"LO2"

For all other converter applications, choose from:

"LO1"

"LO2"

<source> String) Source name from [Source Configuration dialog](#).

Examples

```
SENS:MIX:ROLE:DEV "LO1","LO1Name"  
sense2:mixer:role:device "LO1","LO1Name"
```

Query Syntax SENSe<ch>:MIXer:ROLE:DEvice? <source>

Return Type String

Default Not Applicable

SENSe<ch>:MIXer:SAVE <name>

(Write-only) Saves the settings for the mixer/converter test setup to a mixer attributes file.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<name> Path and file name (including .mxrx extension) to save.

Examples

```
SENSe:MIXer:SAVE "C:/Program Files/Agilent/Network  
Analyzer/Documents/Mixer/MyMixer.mxrx"
```

Default Not Applicable

SENSe<ch>:MIXer:STAGe <n>

(Read-Write) Number of IF stages (LOs) used in the mixer. [See Note](#)

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<n> Number of stages. Choose either **1** or **2**

Examples

```
SENSe:MIX:STAG 2  
SENSe:MIXer:STAGe 1
```

Query Syntax SENSe<ch>:MIXer:STAGe?

Return Type Numeric

Default 1

SENSe<ch>:MIXer:XAXis <char>

(Read-Write) Sets or returns the swept frequency range to display on the X-axis for the IMDx channel.

For FCA and GCX measurements, use [CALC:MIXer:XAXis](#)

Parameters

- <ch> Channel number of the IMDx Converter measurement. If unspecified, value is set to 1.
- <char> Frequency range to display on the X-Axis. NOT case-sensitive. Choose from:
- **INPUT** - Input frequency range
 - **LO_1** - LO frequency range
 - **OUTPUT** - Output frequency range

If the specified frequency range is not swept, the default swept range is used.

Examples

```
SENSe:MIXer:XAXis INPUT  
sense2:mixer:xaxis LO_1
```

Return Type Character

Default Search is made in the following order until a swept range is found:

1. OUTPUT
2. INPUT (If the OUTPUT is fixed)
3. Number of Points (If ALL ranges are fixed)

Last Modified:

25-Nov-2013	Fixed links in tree
6-Oct-2011	Added Point to Normalize
25-Mar-2010	Added Include Phase (9.2)
12-Feb-2010	Added Include Reverse
12-Oct-2009	Clarified Stages
13-Mar-2009	Added roles and X-axis for IMDx
3-Mar-2009	Fixed Mix:Calc for INP and OUTP
24-Feb-2009	Replaced True/False
17-Dec-2008	Fixed map & typo
2-Oct-2008	Clarification of sideband commands
5-May-2008	Added Port Map commands
6-Mar-2008	Added note to page top
23-Jul-2007	Clarified LO Name command

Sense:Mixer:ELO Commands

Allows measurements of a Mixer with an Embedded LO.

```
SENSE:MIXer:ELO
  LO:DELTA
  LO:RESet
  NORMalize:POINT
  STATE
  TUNing
    | IFBW
    | INTerval
    | ITERations
    | MODE
    | RESet
    | SPAN
    | TOLerance
  DIAGnostic
    | CLEar
    | STATus
    | SWEEp
      | COUNT?
      | LO:DELTA?
      | MARKer
        | ANNotation?
        | POSition?
        | STATE?
      | PARAmeter?
    | X
      | ANNotation?
      | START?
      | STOP?
    | Y
      | ANNotation?
```

Click on a [blue](#) keyword to view the command details.

Note: The **Find Now** feature on the Embedded LO dialog is performed remotely by doing a 'Single Sweep' with ELO enabled. [See example program.](#)

- [Example Programs](#)
- [Learn about Embedded LO Settings](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

Note: The Embedded LO **DIAGnostic** commands read data from the various broadband and precise tuning sweeps, similar to the textual and graphical data that are available in the user interface.

SENSe<ch>:MIXer:ELO:LO:DELTA <num>

(Read-Write) Sets and returns LO Frequency Delta. There is usually no need to set this value. Read this value to determine the difference between the LO Frequency that is stated in the Mixer dialog box and the last measured LO Frequency.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <num> LO Frequency delta in Hertz.

Examples `SENS:MIX:ELO:LO:DELTA 10.3`

Query Syntax `SENSe<ch>:MIXer:ELO:LO:DELTA?`

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:ELO:LO:RESet

(Write-only) Resets the LO Delta Frequency to 0 (zero).

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1

Examples `SENS:MIX:ELO:LO:RES`
`sense2:mixer:elo:lo:reset`

Query Syntax Not Applicable

Default Not Applicable

SENSe<ch>:MIXer:ELO:NORMalize:POINT <num>

(Read-Write) Sets and returns the sweep data point around which to perform broadband and precise tuning.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <num> Mixer Sweep data point. Choose a data point number, between 1 and the max number of data points in the sweep, that has the least amount of expected noise.

Examples

```
SENS:MIX:ELO:LO:NORM:POIN 200  
sense2:mixer:elo:normalize:point 101
```

Query Syntax SENSE<ch>:MIXer:ELO:NORMAlize:POINT?

Return Type Numeric

Default Center point in the sweep span

SENSE<ch>:MIXer:ELO:STATE <bool>

(Read-Write) Sets and returns the ON |OFF state of Embedded LO.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <bool> ON | OFF state. Choose from
 - 0 - Embedded LO OFF
 - 1 - Embedded LO ON

Examples

```
SENS:MIX:ELO:STAT 1  
sense2:mixer:elo:state 0
```

Query Syntax SENSE<ch>:MIXer:ELO:STATE?

Return Type Boolean

Default OFF

SENSE<ch>:MIXer:ELO:TUNing:IFBW <num>

(Read-Write) Sets and returns the IF Bandwidth for Broadband and Precise tuning sweeps.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <num> IF Bandwidth

Examples

```
SENS:MIX:ELO:TUN:IFBW 10kHz  
sense2:mixer:elo:tuning:ifbw 20e3
```

Query Syntax SENSE<ch>:MIXer:ELO:TUNing:IFBW?

Return Type Numeric

Default 30kHz

SENSE<ch>:MIXer:ELO:TUNing:INTERval <num>

(Read-Write) Sets and returns how often a tuning sweep is performed.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <num> Tuning sweep interval

Examples

```
SENS:MIX:ELO:TUN:INT 2  
sense2:mixer:elo:tuning:interval 1
```

Query Syntax SENSE<ch>:MIXer:ELO:TUNing:INTERval?

Return Type Numeric

Default 1

SENSE<ch>:MIXer:ELO:TUNing:ITERations <num>

(Read-Write) Sets and returns the maximum number of tuning iterations to achieve the precise tolerance.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <num> Number of tuning iterations. Choose a number between 1 and 100.

Examples

```
SENS:MIX:ELO:TUN:ITER 5  
sense2:mixer:elo:tuning:iterations 3
```

Query Syntax SENSE<ch>:MIXer:ELO:TUNing:ITERations?

Return Type Numeric

Default 5

SENSE<ch>:MIXer:ELO:TUNing:MODE <char>

(Read-Write) Sets and returns the method used to determine the embedded LO Frequency.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <char> Tuning mode. Choose from:

BROadband Both broadband and precise tuning

PRECise Precise tuning only

NONE No tuning; just apply the LO Frequency Delta value.

Examples

```
SENS:MIX:ELO:TUN:MODE BRO  
sense2:mixer:elo:tuning:mode precise
```

Query Syntax SENSE<ch>:MIXer:ELO:TUNing:MODE?

Return Type Character

Default BROadband

SENSE<ch>:MIXer:ELO:TUNing:RESet

(Write-only) Resets the tuning parameters to their default values.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

Examples

```
SENS:MIX:ELO:TUN:RES  
sense2:mixer:elo:tuning:reset
```

Query Syntax Not Applicable

Default Not Applicable

SENSe<ch>:MIXer:ELO:TUNing:SPAN <num>

(Read-Write) Sets and returns the frequency span for the broadband tuning sweep.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<num> Broadband frequency span in Hz.

Examples

```
SENS:MIX:ELO:TUN:SPAN 1e6  
sense2:mixer:elo:tuning:span 1mhz
```

Query Syntax SENSe<ch>:MIXer:ELO:TUNing:SPAN?

Return Type Numeric

Default 3 MHz

SENSe<ch>:MIXer:ELO:TUNing:TOLerance <num>

(Read-Write) Sets and returns the tuning tolerance for precise tuning.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<num> Tuning tolerance in Hz. Choose a number between .001 and 1e3.

Examples

```
SENS:MIX:ELO:TUN:TOL .5  
sense2:mixer:elo:tuning:tolerance 1
```

Query Syntax SENSe<ch>:MIXer:ELO:TUNing:TOLerance?

Return Type Numeric

Default 1 Hz

SENSe<ch>:MIXer:ELO:DIAGnostic:CLEar

(Write-only) Clears current diagnostic information.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

Examples

```
SENS:MIX:ELO:DIAG:CLEAr  
sense2:mixer:elo:diagnostic:clear
```

Query Syntax Not Applicable

Default Not Applicable

SENSe<ch>:MIXer:ELO:DIAGnostic:STATus?

(Read-only) Returns a string that describes the result of the last tuning sweeps.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

Examples

```
SENS:MIX:ELO:DIAG:STAT?  
sense2:mixer:elo:diagnostic:status
```

Return Type String

Default Not Applicable

SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep:COUNT?

(Read-only) Returns the number of tuning sweeps used for the latest embedded LO measurement.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

Examples

```
SENS:MIX:ELO:DIAG:SWEep:COUNT  
sense2:mixer:elo:diagnostic:sweep:count?
```

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:LO:DELTA?

(Read-only) Returns the LO frequency delta from the specified tuning sweep.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Tuning sweep number. Use [SENS:MIX:ELO:DIAG:SWE:COUNt?](#) to find the number of sweeps taken.

Examples

```
SENS:MIX:ELO:DIAG:SWE2:LO:DELt?  
sense2:mixer:elo:diagnostic:sweep1:lo:delta?
```

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:MARKer:ANNotation?

(Read-only) Returns the Y-axis marker value from the specified tuning sweep. This command assumes that a marker was used. Use SENS:MIX:ELO:DIAG:SWE:MARK:STATe? to confirm if a marker was used for the tuning sweep.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Tuning sweep number. Use [SENS:MIX:ELO:DIAG:SWE:COUNt?](#) to find the number of sweeps taken.

Examples

```
SENS:MIX:ELO:DIAG:SWE2:MARKer:ANN?  
sense2:mixer:elo:diagnostic:sweep1:marker:annotation?
```

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:MARKer:POSition?

(Read-only) Returns the X-axis marker position from the specified tuning sweep.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Tuning sweep number. Use [SENS:MIX:ELO:DIAG:SWE:COUNT?](#) to find the number of sweeps taken.

Examples

```
SENS:MIX:ELO:DIAG:SWE2:MARK:POS?  
sense2:mixer:elo:diagnostic:sweep1:marker:position?
```

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:MARKer:STATe?

(Read-only) Returns whether or not a marker was used for the specified tuning sweep.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Tuning sweep number. Use [SENS:MIX:ELO:DIAG:SWE:COUNT?](#) to find the number of sweeps taken.

Examples

```
SENS:MIX:ELO:DIAG:SWE2:MARK:POS?  
sense2:mixer:elo:diagnostic:sweep1:marker:position?
```

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:PARAmeter?

(Read-only) Returns the name of the parameter of the specified tuning sweep.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Tuning sweep number. Use [SENS:MIX:ELO:DIAG:SWE:COUNT?](#) to find the number of sweeps taken.

Examples

```
SENS:MIX:ELO:DIAG:SWE2:PAR?  
sense2:mixer:elo:diagnostic:sweep1:parameter?
```

Return Type String - either "VC21" or "B,1"

Default Not Applicable

SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:TITLe?

(Read-only) Returns the title of the specified tuning sweep.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Tuning sweep number. Use [SENS:MIX:ELO:DIAG:SWE:COUNT?](#) to find the number of sweeps taken.

Examples

```
SENS:MIX:ELO:DIAG:SWE2:TITL?  
sense2:mixer:elo:diagnostic:sweep1:title?
```

Return Type String

Default Not Applicable

SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:X:ANNotation?

(Read-only) Returns the X-Axis annotation of the specified tuning sweep.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Tuning sweep number. Use [SENS:MIX:ELO:DIAG:SWE:COUNT?](#) to find the number of sweeps taken.

Examples

```
SENS:MIX:ELO:DIAG:SWE2:X:ANN?  
sense2:mixer:elo:diagnostic:sweep1:x:annotation?
```

Return Type String - either "Hz" or "s"

Default Not Applicable

SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:X:STARt?

(Read-only) Returns the start value of the specified tuning sweep.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Tuning sweep number. Use [SENS:MIX:ELO:DIAG:SWE:COUNT?](#) to find the number of sweeps taken.

Examples

```
SENS:MIX:ELO:DIAG:SWE2:X:STAR?  
sense2:mixer:elo:diagnostic:sweep1:x:start?
```

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:X:STOP?

(Read-only) Returns the stop value of the specified tuning sweep.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Tuning sweep number. Use [SENS:MIX:ELO:DIAG:SWE:COUNt?](#) to find the number of sweeps taken.

Examples

```
SENS:MIX:ELO:DIAG:SWE2:X:STOP?  
sense2:mixer:elo:diagnostic:sweep1:x:stop?
```

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:Y:ANNotation?

(Read-only) Returns Y-axis annotation value of the specified tuning sweep.

Parameters

- <ch> Any existing channel number. If unspecified, value is set to 1
- <n> Tuning sweep number. Use [SENS:MIX:ELO:DIAG:SWE:COUNt?](#) to find the number of sweeps taken.

Examples

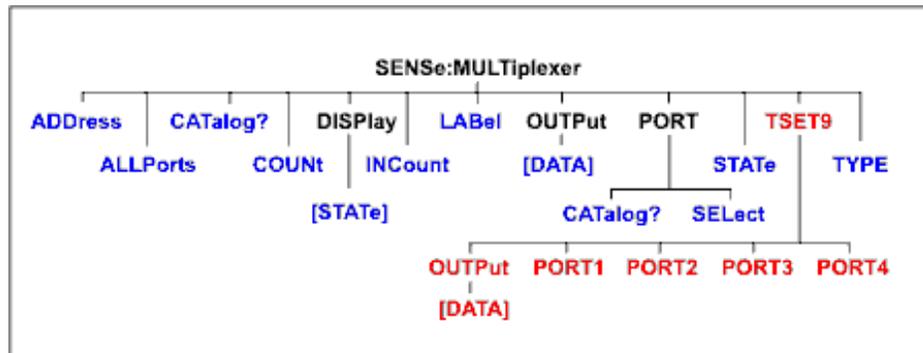
```
SENS:MIX:ELO:DIAG:SWE2:Y:ANN?  
sense2:mixer:elo:diagnostic:sweep1:y:annotation?
```

Return Type String - either "U" or "Phase"

Default Not Applicable

Sense:Multiplexer Commands

Controls External Test Sets (N44xx, E5091A, "Z", and "H" series).



Click on a [blue](#) keyword to view the command details.

Red commands are [superseded](#).

See Also

- [See an example program](#) using these commands.
- [Learn about External Test Set Control](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

SENSe:MULTiplexer<id>:ADDDress <address>

(Read-Write) Sets and returns the address for the external test set at the specified ID. This command should be immediately preceded by the [SENSe:MULT:TYPE](#) command.

Parameters

<id> Id of the external test set. If unspecified, Id is assumed to be 1. Must be previously set by the [SENSe:MULT:TYPE](#) command.

<address> Integer The test set address.

- For a GPIB test set (N44xx and some specials), this is the GPIB address.
- For a test set I/O test set (some specials), it is the position of the test set in the chain (starting at 0).
- For USB test sets (E5091A), the address is set by DIP switches on the rear panel.

Examples `SENS:MULT1:TYPE "Z5623A_K66" ' use K66 test set, and reference it through ID 1`
`SENS:MULT1:ADDR 0 ' first test set in sequence`
`' All subsequent commands using SENS:MULT1 will refer to this test set`

Query Syntax `SENSe:MULTiplexer<id>:ADDRess?`

Return Type Numeric

Default Not Applicable

SENSe<cnum>:MULTiplexer<id>:ALLPorts <string>

(Read-Write) Sets or gets the port selections for all available ports on the specified channel.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1.

<id> Id of the external test set. If unspecified, Id is assumed to be 1. Must be previously set by the [SENSe:MULT:TYPE](#) command.

<string> Comma-separated list of port selections, one for each port. Each port selection must correspond to one of the values returned by [SENS:MULT:PORT:CAT?](#).

Do NOT include + and - .

Examples `' for channel 5 and test set 1, set port 1 to T1,`
`' port 2 to A, port 3 to R2+, port 4 to R3-.`
`SENS5:MULT1:ALLP "T1,A,R2,R3 "`

Query Syntax `SENSe<cnum>:MULTiplexer<id>:ALLPorts?`

Return Type STRING

Default Not Applicable

SENSe:MULTiplexer:CATalog?

(Read-Only) Returns a comma-separated list of the external test sets models that are currently supported. Choose one of these items to send [SENS:MULT1:TYPE](#).

Examples `SENS:MULT:CAT?`

Return Type String

Default Not Applicable

SENSe:MULTiplexer<id>:COUNT?

(Read-Only) Returns the total number of ports of the specified test set.

Returns 0 if no test set is connected (GPIB test sets only).

Parameters

<id> Id of the external test set. If unspecified, Id is assumed to be 1. Must be previously set by the [SENSe:MULT:TYPE](#) command.

Examples

```
SENS:MULT1:COUN?  
sense:multiplexer2:count?
```

Return Type Numeric

Default Not Applicable

SENSe:MULTiplexer<id>:DISPlay[:STATe] <bool>

(Read-Write) Turns ON and OFF the display of the test set control status bar. This status bar indicates the test set that is being controlled and the current port mappings. This setting is turned ON automatically when the test set is enabled.

Parameters

<id> Id of the external test set. If unspecified, Id is assumed to be 1. Must be previously set by the [SENSe:MULT:TYPE](#) command.

<bool> ON(1) Turns ON the display.

OFF (0) Turns OFF the display.

Examples

```
SENS:MULT1:DISP 1  
sense:multiplexer2:display:state on
```

Query Syntax SENSe:MULTiplexer<id>:DISPlay[:STATe]?

Return Type Boolean

Default OFF (0)

SENSe:MULTiplexer<id>:INCount?

(Read-Only) Returns the number of input ports for the specified test set.

- For test sets such as the E5091A that do NOT use jumper cables to route the stimulus and response signals, this command returns the number of test set ports that can be connected to the PNA.
- For test sets that DO use jumper cables to route the stimulus and response signals, such as the N44xx, the return value is not valid.

Parameters

<id> Id of the external test set. If unspecified, Id is assumed to be 1. Must be previously set by the [SENSe:MULT:TYPE](#) command.

Examples

```
SENS3:MULT1:INC? ' returns the number of input ports for test set 1 on channel 3
```

Return Type Numeric

Default Not Applicable

SENSe<cnum>:MULTiplexer:LABel <string>

(Read-Write) Sets and returns the display label for the testset on the specified channel. The label appears in a status bar at the bottom of the PNA display when [SENS:MULT:DISP](#) is set to ON.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1.

<string> Display label text.

Examples

```
SENS3:MULT:LAB 'High-power output'
```

Query Syntax SENSe<cnum>:MULTiplexer:LABel?

Return Type String

Default Not Applicable

SENSe<cnum>:MULTiplexer<id>:OUTPut[:DATA] <num>

(Read-Write) Sets or returns the control line value for the specified channel.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1.
- <id> Id of the external test set. If unspecified, Id is assumed to be 1. Must be previously set by the [SENSe:MULT:TYPE](#) command.
- <numr> An integer specifying the decimal value of the control line. Values are obtained by adding weights from the following table that correspond to individual lines.

Line	Weight
1	1
2	2
3	4
4	8
5	16
6	32
7	64
8	128

Note:

- The E5091A interprets SENS:MULT1:OUTP 0 as all lines OFF.
- All "Z" and "H" series test sets interpret SENS:MULT1:OUTP 0 as all lines ON.

Refer to your test set documentation for setting control line values.

Examples `SENS3:MULT1:OUTP 48` 'For Z5623A K64, lines 5 and 6 are OFF; all other lines are set to ON state.'

Query Syntax `SENSe<cnum>:MULTiplexer<id>:OUTPut[:DATA]?`

Return Type Numeric

Default Not Applicable

[SENSe:MULTiplexer<id>:PORT<pnum>:CATalog?](#)

(Read-Only) Returns a comma-separated list of valid port selections for the specified port.

Parameters

- <id> Id of the external test set. If unspecified, Id is assumed to be 1. Must be previously set by the [SENSe:MULT:TYPE](#) command.
- <pnum> Integer - Input port number for which to return valid Output port selections. Read the number of input ports for the test set using [SENS:MULT:INCount?](#)

Examples `SENS:MULT1:PORT3:CAT? ' returns the valid port selections for port 3`

Return Type String

Default Not Applicable

SENSe<cnum>:MULTiplexer<id>:PORT<pnum>:SElect <string>

(Write-Only) Sets and returns a port mapping for a single port. If this command creates a conflict with an existing port, the PNA will resolve the conflict.

Note: This command is not supported for the Z5623AK44.

Parameters

- <cnum> Channel number of the measurement. If unspecified, value is set to 1.
- <id> Id of the external test set. If unspecified, Id is assumed to be 1. Must be previously set by the [SENSe:MULT:TYPE](#) command.
- <pnum> Integer - Logical port number.
- <string> Physical port number.

Examples `SENS:MULT1:PORT3:SEL "4" 'sets logical port 3 to physical port 4.`

Return Type String

Default Not Applicable

SENSe:MULTiplexer<id>:STATe <bool>

(Read-Write) Enables and disables (ON/OFF) the port mapping and control line output of the specified test set.

If the specified test set is not connected or not ON, then setting State ON will report an error. All other properties can be set when the test set is not connected.

When this command is set to ON, then the display of the test set status bar ([SENS:MULT:DISP](#)) is also set to ON.

Parameters

- <id> Id of the external test set. If unspecified, Id is assumed to be 1. Must be previously set by the [SENSe:MULT:TYPE](#) command.
- <bool> ON(1) Enables test set control.
OFF (0) Disables test set control.

Examples

```
SENS:MULT1:STAT 1  
sense2:multiplexer2:state on
```

Query Syntax SENSE<cnum>:MULTiplexer<id>:STATe?

Return Type Boolean

Default OFF (0)

SENSe<cnum>:MULTiplexer<id>:TSET9:OUTPut[:DATA] <data> Superseded

Note: This command is replaced with [SENS:MULT:OUTP](#)

(Read-Write) Sets the control lines of the specified E5091A. Control lines, provided through a E5091A front panel connector, are used to control external equipment such as a part handler. See your E5091A documentation to learn more about control lines.

Parameters

- <cnum> Channel number of the measurement. If unspecified, value is set to 1.
- <id> Id of the E5091A test set. Choose from 1 or 2. [Learn how to set ID value.](#)
- <data> Data value used to set control lines. Values are obtained by adding weights from the following table that correspond to individual lines. HIGH =1; LOW=0.

Line	Weight
1	1
2	2
3	4
4	8
5	16
6	32
7	64
8	128

0 - Sets all lines low

255 - Sets all lines high

Examples

'The following sets line 3 and 4 high. All other lines low.'

```
SENS:MULT1:TSET9:OUTP 12
```

Query Syntax SENSE<cnum>:MULTiplexer<id>:TSET9:OUTPut[:DATA]?

Return Type Numeric

Default 0

SENSe<cnum>:MULTiplexer<id>:TSET9:PORT1 <char> Superseded

Note: This command is replaced with [SENS:MULT:ALLPorts](#) which sets ALL ports to the specified outputs.

(Read-Write) Switches Port 1 of the specified E5091A to one of the available outputs.

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1.

<id> Id of the E5091A test set. Choose from 1 or 2. [Learn how to set ID value.](#)

<char> Output port to be switched to. Choose from:

A

T1 - (If Port 2 already is connected to T1, then Port 2 will be switched to T2.)

Examples

```
SENS:MULT1:TSET9:PORT1 A
```

Query Syntax SENSE<cnum>:MULTiplexer<id>:TSET9:PORT1?

Return Type Character

Default A

SENSe<num>:MULTiplexer<id>:TSET9:PORT2 <char> Superseded

Note: This command is replaced with [SENS:MULT:ALLPorts](#) which sets ALL ports to the specified outputs.

(Read-Write) Switches Port 2 of the specified E5091A to one of the available outputs.

Parameters

<num> Any existing channel number; if unspecified, value is set to 1.

<id> Id of the E5091A test set. Choose from 1 or 2. [Learn how to set ID value.](#)

<char> Output port to be switched to. Choose from:

T1 - If Port 1 already is connected to T1, then Port 1 will be switched to A.

T2

Examples

```
SENS:MULT1:TSET9:PORT2 T2
```

Query Syntax SENSe<num>:MULTiplexer<id>:TSET9:PORT2?

Return Type Character

Default T1

SENSe<num>:MULTiplexer<id>:TSET9:PORT3 <char> Superseded

Note: This command is replaced with [SENS:MULT:ALLPorts](#) which sets ALL ports to the specified outputs.

(Read-Write) Switches Port 3 of the specified E5091A to one of the available outputs.

Parameters

<num> Any existing channel number; if unspecified, value is set to 1.

<id> Id of the E5091A test set. Choose from 1 or 2. [Learn how to set ID value.](#)

<char> Output port to be switched to. Choose from:

R1 (R1+)

R2 (R2+)

R3 (R3+) If option 007 (7port), R2 is selected.

Examples `SENS:MULT1:TSET9:PORT3 R2`

Query Syntax `SENSe<cnum>:MULTiplexer<id>:TSET9:PORT3?`

Return Type Character

Default R1

SENSe<cnum>:MULTiplexer<id>:TSET9:PORT4 <char> Superseded

Note: This command is replaced with [SENS:MULT:ALLPorts](#) which sets ALL ports to the specified outputs.

(Read-Write) Switches Port 4 of the specified E5091A to one of the available outputs.

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1.

<id> Id of the E5091A test set. Choose from 1 or 2. [Learn how to set ID value.](#)

<char> Output port to be switched to. Choose from:

R1 (R1-)

R2 (R2-)

R3 (R3-) If option 007 (7port), R2 is selected.

Examples `SENS:MULT1:TSET9:PORT4 R2`

Query Syntax `SENSe<cnum>:MULTiplexer<id>:TSET9:PORT4?`

Return Type Character

Default R1

SENSe:MULTiplexer<id>:TYPE <name>

(Read-Write) Loads a configuration file for the specified type of external test set.

This command should be immediately followed by the [SENSe:MULT:ADDRess](#) command.

Parameters

<name> String The name of the type of test set. Must be one of the items in the list returned by the [SENSe:MULT:CATalog?](#) query.

<id> Id of the external test set. Set by this command. Use consecutive values starting at 1.

Examples

```
SENS:MULT1:TYPE "Z5623AK66" ' use K66 test set, and reference  
it through ID 1
```

Query Syntax SENSE:MULTiplexer<id>:TYPE?

Return Type String

Default Not Applicable

Last Modified:

11-Jan-2011 Minor edits

17-Aug-2007 Modified Data command for differences in active high and low

SENSe:NOISe Commands

Controls the Noise Figure / NFX configuration and calibration.

SENSe:NOISe:

AVERage <num>

| **STATe** <bool>

BWIDth <num>

CALibration:

| **METHod** <string>

| **RMETHod** <string>

ENR:FILEname <string>

GAIN <num>

IMPedance:COUNT <num>

PMAP <in>,<out>

| **INPut?**

| **OUTPut?**

PULL[:STATe] <bool>

RECeiver <char>

SNP? <string>

| **SAVE** <string>

SOURce:

| **CKIT** <string>

| **CONNector** <string>

TEMPerature:AMBient <num>

TUNer:

| **ID** <string>

| **INPut** <string>

| **ORient[:STATe]**

| **OUTPut** <string>

Click on a [blue](#) keyword to view the command details.

Other Noise Figure SCPI commands

The calibration commands listed in this topic are supplemental to the [Guided Cal commands](#).

- [CALC:CUSTom:DEFine](#) - creates a noise figure measurement.
- [CONTrol:NOISe:SOURce](#) or [OUTPut:MANual:NOISe\[:STATe\]](#) - turns the Noise Source ON and OFF.
- [MMEMory:LOAD:ENR](#) and [MMEM:STORe:ENR](#) - load and save ENR files.
- [SENSe:PATH:CONF:ELEMent\[:STATe\]](#) - sets the port 1 and port 2 noise switches.
- [SENS:CORR:ENR:CAL](#):- manage ENR data - usually not necessary.
- [SYST:PREF:ITEM:SWIT:DEF](#) - Sets the default setting of the Noise Tuner switch
- [SENS:CORR:NOISe](#) commands - noise calibration
- [SENS:CORR:Guided](#) commands - performs most of noise cal.

See Also

- **Examples:**
 - [Create and Cal a Noise Figure Measurement](#)
 - [Create and Cal an NFX Measurement](#)
- [Learn about Noise Figure Application](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

SENSe<ch>:NOISe:AVERAge[:COUNT] <num>

(Read-Write) Set and read the averaging factor for the noise receiver. [Learn more](#)

Parameters

- <ch> Noise Figure channel number. If unspecified, value is set to 1.
- <num> Averaging value. Choose any number from 1 to 99.

Examples

```
SENS:NOIS:AVER 20  
sense:noise:average:count 10
```

Query Syntax SENSE:NOISe:AVERAge[:COUNT]?

Return Type Numeric

Default 1

SENSe<ch>:NOISe:AVERAge:STATe <bool>

(Read-Write) Turns noise averaging ON and OFF.

Parameters

- <ch> Noise Figure channel number. If unspecified, value is set to 1.
- <bool> Averaging state. Choose from
 - 0 - Noise averaging OFF
 - 1 - Noise averaging ON

Examples

```
SENS:NOIS:AVER:STAT 0  
sense:noise:average:state 1
```

Query Syntax SENSE:NOISe:AVERAge:STATe?

Return Type Boolean

Default 0 - OFF

SENSe<ch>:NOISe:BWIDth[:RESolution] <num>

(Read-Write) Set and read the bandwidth of the noise receiver. [Learn more](#)

Parameters

- <ch> Noise Figure channel number. If unspecified, value is set to 1.
<num> Bandwidth value. Choose from:

For [Sens:Noise:Receiver](#) = **NOISe** (Opt 029) choose from: 800 KHz, 2 MHz, 4 MHz, 8 MHz, or 24 MHz or the numerical equivalent, such as 8e6 and so forth.

For [Sens:Noise:Receiver](#) = **NORMal** (Opt 028) choose from: 720 kHz or 1.2 MHz

If the value does not match one of these, it is rounded up to the next valid bandwidth value.

Examples

```
SENS:NOIS:BWID 2e6  
sense:noise:bandwidth:resolution 8mhz
```

Query Syntax SENSE:NOISe:BWIDth[:RESolution]?

Return Type Numeric

Default 4 MHz for Noise Receiver

1.2 MHz for Normal Receiver

SENSe<ch>:NOISe:CALibration:METHOD <string>

(Read-Write) Set and read the method for performing a calibration on a noise channel.

Parameters

- <ch> Noise Figure channel number. If unspecified, value is set to 1.
<string> Calibration method. NOT case-sensitive. Choose from:

- "VectorFull" or "Vector"
- "SParameter" (Not available for NFX measurements)
- "ScalarFull" or "Scalar"

Examples

```
SENS:NOIS:CAL:METHOD "Vector"  
sense:noise:calibration:method "SParameter"
```

Query Syntax SENSE:NOISe:CALibration:METHOD?

Return Type String

Default "VectorFull"

SENSe<ch>:NOISe:CALibration:RMETHOD <string>

(Read-Write) Set and read the method used to characterize the noise receivers. [Learn more.](#)

Parameters

- <ch> Noise Figure channel number. If unspecified, value is set to 1.
- <string> Receiver characterization method. NOT case-sensitive. Choose from:
- "NoiseSource" - Use a noise source. This selection is NOT allowed when a standard PNA receiver is used as the noise receiver ([SENS:NOIS:REC NORM](#)).
 - "PowerMeter" - Use a power meter. **NOTE:** This selection is NOT allowed when the [Noise Bandwidth](#) is 8 MHz or 24 MHz.

Examples

```
SENS:NOIS:CAL:RMET "PowerMeter"  
sense:noise:calibration:rmethod "noisesource"
```

Query Syntax SENSE:NOISe:CALibration:RMETHOD?

Return Type String

Default "NoiseSource"

SENSe<ch>:NOISe:ENR:FILENAME <string>

(Read-Write) Set and read the path and name of the ENR file associated with the noise source.

Parameters

- <ch> Noise Figure channel number. If unspecified, value is set to 1.
- <string> Full path, filename, and extension of the ENR file.

Examples

```
SENS:NOIS:ENR:FIL "c:/ProgramFiles/Agilent/Network  
Analyzer/Documents/ENR/346C.enr"  
sense:noise:enr:filename "c:/ProgramFiles/Agilent/Network  
Analyzer/Documents/ENR/346C.enr"
```

Query Syntax SENSE:NOISe:ENR:FILENAME?

Return Type String

Default Not applicable

SENSe<ch>:NOISe:GAIN <num>

(Read-Write) Set and read the amount of gain for the noise receiver. This setting is NOT used when [Sens:Noise:Receiver](#) = **NORMal** (Opt 028)

Parameters

- <ch> Noise Figure channel number. If unspecified, value is set to 1.
- <num> Gain value. Choose from:
- **0** - Low gain; select if the gain of your DUT is relatively high (>35 dB).
 - **15** - Medium gain; select if the gain of your DUT is about average (20 dB to 45 dB)
 - **30** - High gain; select if the gain of your DUT is relatively low (<30 dB).

[Learn more about Noise Receiver Gain setting.](#)

If the value does not match one of these, it is rounded up to the next legal value.

Examples

```
SENS:NOIS:GAIN 15  
sense:noise:gain 0
```

Query Syntax SENSe:NOISe:GAIN?

Return Type Numeric

Default 30

SENSe<ch>:NOISe:IMPedance:COUNt <num>

(Read-Write) Sets the number of impedance states to use during calibrated measurements.

Parameters

- <ch> Noise Figure channel number. If unspecified, value is set to 1.
- <num> Number of impedance states to use. Choose between 4 and the maximum number allowed by the noise tuner device. The more states that are used, the more accurate, and slower, the measurement. If the specified number exceeds the capability of the device, the measurement will use the maximum number of states the device allows.

Examples

```
SENS:NOIS:IMP:COUN 5  
sense:noise:impedance:count 7
```

Query Syntax SENSe:NOISe:IMPedance:COUNt?

Return Type Numeric

Default 4

SENSe<ch>:NOISe:PMAP <in>,<out>

(Write-only) Set and read the DUT-to-PNA port mapping for Noise Figure. Port mapping is allowed ONLY when [SENS:NOIS:REC](#) is set to **NORM** (standard PNA-X receiver).

Parameters

<ch> Any existing NF or NFX channel. If unspecified, value is set to 1.

<in> PNA port which is connected to the DUT input.

<out> PNA port which is connected to the DUT output.

Examples

```
SENS:NOIS:PMAP 1,2
```

```
sense:noise:pmap 2,1
```

[See example program](#)

Query Syntax Not Applicable

Default 1,2

SENSe<ch>:NOISe:PMAP:INPut?

(Read-only) Read the PNA port number to be connected to the DUT Input.

Use [SENS:NOISe:PMap](#) to set the port mapping.

Parameters

<ch> Any existing NF or NFX channel. If unspecified, value is set to 1.

Examples

```
SENS:NOIS:PMAP:INP?
```

```
sense:noise:pmap:input?
```

Return Type Numeric

Default 1

SENSe<ch>:NOISe:PMAP:OUTPut?

(Read-only) Read the PNA port number to be connected to the DUT Output.

Parameters

<ch> Any existing NF or NFX channel. If unspecified, value is set to 1.

Examples

```
SENS:NOIS:PMAP:OUTP?  
sense:noise:pmap:output?
```

Return Type Numeric

Default 2

SENSe<ch>:NOISe:PULL[:STATe] <bool>

(Read-Write) Enables and disables the use of source pull technique to compute S22. [Learn more.](#)

Parameters

<ch> Noise Figure channel number. If unspecified, value is set to 1.

<bool> Source pull technique state. Choose from:

OFF or **0** - Disable use of source pull technique.

ON or **1** - Enable use of source pull technique.

Examples

```
SENS:NOIS:PULL 0  
sense2:noise:pull:state ON
```

Query Syntax SENSe:NOISe:PULL[:STATe]?

Return Type Boolean

Default 0 - OFF

SENSe<ch>:NOISe:RECeiver <char>

(Read-Write) Sets and returns the noise receiver to use for noise measurements.

Parameters

- <ch> Noise Figure channel number. If unspecified, value is set to 1.
- <char> Noise receiver. Choose from:
 - NORMal** The standard PNA receiver. (Opt 028 and Opt 029)
 - NOISe** The low-noise receivers. (Opt 029 only)

Examples

```
SENS:NOIS:REC NORM  
sense2:noise:receiver noise
```

Query Syntax SENSE:NOISe:RECEiver?

Return Type Character

Default NOISe

SENSe<ch>:NOISe:SNP? [string]

(Read-Only) Returns S-parameter and noise parameter data for vector noise figure measurements.

Noise parameters are NOT valid for NFX or Scalar noise figure measurements. Learn more about [noise parameters](#).

Parameters

- <ch> Noise Figure channel number. If unspecified, value is set to 1.
- [string] Optional parameter. Choose "**NoiseParameter**" - Noise parameter data.
If unspecified, only S-parameter data is saved.

Examples

```
SENS:NOIS:SNP?  
sense2:noise:snp? "NoiseParameter"
```

Return Type Comma-separated values

Data is returned in this order:

1. <all frequencies | point number>
2. <real S11> <imag S11> <real S21> <imag S21> <real S12> <imag S12>
<real S22> <imag S22>

Then if noise parameters are specified:

3. <NFMin dB> <mag GammaOpt> <phase GammaOpt> <Rn/Z0>

The data display format depends on [MMEM:STOR:TRAC:FORM:SNP](#)

Default Not Applicable

SENSe<ch>:NOISe:SNP:SAVE <filename>, [data]

(Write-only) Saves the S-parameters and vector noise parameters to an S2P file. For NFX channels, mixer setup information is included as comments at the beginning of the file.

Learn more about [noise parameters](#).

The format of the snp data is set with [MMEM:STOR:TRAC:FORM:SNP](#).

The following is sample data for two data points from a Noise Figure measurement:

```
! Agilent Technologies,N5242A,USxxxxxxx,A.09.85
! pnan-nn Thu Nov 01 12:26:27 2012
# HZ S MA R 50
!freq (Hz) S11M      S11A      S21M      S21A      S12M      S12A      S22M
      S22A
2000000000 9.038147e-001 6.241193e+001 5.855965e+000 -6.116778e+001 2.232653e-002 -
1.475392e+002 5.275644e-001 1.750775e+002
8000000000 6.951366e-001 -1.458202e+002 5.307699e+000 -7.055212e+001 7.838612e-002 -
1.460951e+002 3.986142e-001 -2.226317e+001
! Noise Parameters
!freq (Hz) NFMin(dB)  Rho_opt(Mag)  Rho_opt(deg)  Rn/Z0
2000000000 1.251697e+000 2.172018e-001 -8.765875e+001 1.806663e-001
8000000000 1.583849e+000 2.015185e-001 1.029875e+002 1.320403e-001
```

Parameters

- <ch> Noise Figure channel number. If unspecified, value is set to 1.
- <filename> Path (optional), filename and suffix of location to store SNP data. If path is not specified, the current path is used.

[data] String. Optional parameter. Choose **"NoiseParameter"** - Noise parameter data.

If unspecified, only S-parameter data is saved.

Examples

```
SENS:NOIS:SNP:SAVE "MySparams.s2p"  
  
sense2:noise:snp:save "C:/Program Files/Agilent/Network  
Analyzer/Documents/MyNoiseParams.s2p", "NoiseParameter"
```

Query Syntax Not Applicable

Default Not Applicable

SENSe<ch>:NOISe:SOURce:CKIT <string>

(Read-Write) Set and read the Cal Kit that will be used for the Noise Source adapter.

An adapter is always necessary to connect a 346C Noise Source to the PNA port 2. Select a Cal Kit that is the same type and gender as the noise source connector.

If the Noise Source mates directly to PNA port 2, then set this type to "None".

Parameters

- <ch> Noise Figure channel number. If unspecified, value is set to 1.
- <string> Cal Kit. Case sensitive.

To read possible cal kit strings for the adapter:

- Change the port connector type to that of the noise source using:
[SENS:CORR:COLL:GUID:CONN:PORT<n>](#)
- Then read the possible cal kit strings for that port using:
[SENS:CORR:COLL:GUID:CKIT:PORT<n>:CAT?](#)

Examples

```
SENS:NOIS:SOUR:CKIT "N4691-60004 ECal"  
  
sense:noise:source:ckit "
```

Query Syntax SENSe:NOISe:SOURce:CKIT?

Return Type String

Default Not applicable

SENSe<ch>:NOISe:SOURce:CONNector <string>

(Read-Write) Set and read the Noise Source connector type and gender. The Agilent 346C has an "APC 3.5 male" connector.

Parameters

- <ch> Noise Figure channel number. If unspecified, value is set to 1.
- <string> Noise source connector type and gender. Case sensitive.

Use [SENS:CORR:COLL:GUID:CONN:CAT?](#) to read possible connector strings.

Examples

```
SENS:NOIS:SOUR:CONN "APC 3.5 male"  
sense:noise:source:connector "APC 3.5 female"
```

Query Syntax SENSE:NOISE:SOURce:CONNector?

Return Type String

Default Not applicable

SENSe<ch>:NOISE:TEMPerature:AMBient <num>

(Read-Write) Sets the temperature at which the current noise measurement is occurring. [Learn more](#)

Parameters

- <ch> Noise Figure channel number. If unspecified, value is set to 1.
- <num> Ambient temperature in Kelvin.

Examples

```
SENS:NOIS:TEMP:AMB 292  
sense:noise:temperature 289
```

Query Syntax SENSE:NOISE:TEMPerature:AMBient?

Return Type Numeric

Default 295

SENSe<ch>:NOISE:TUNer:ID <string>

(Read-Write) Set and read the identity of the noise tuner. This is an ECal model and serial number string. To read the identities of the connected ECal modules, use [SENSe:CORRection:CKIT:ECAL:LIST?](#) and [SENSe:CORRection:CKIT:ECAL<mod>:INFORMATION?](#)

Parameters

- <ch> Noise Figure channel number. If unspecified, value is set to 1.
- <string> ECal model and serial number string. The ECal module must be connected when this command is sent.

Examples

```
SENS:NOIS:TUN:ID "N4691-60004 ECal 02822"  
sense:noise:tuner:id "N4691-60004 ECal 02822"
```

Query Syntax SENSE:NOISe:TUNer:ID?

Return Type String

Default Not applicable

SENSe<ch>:NOISe:TUNer:INPut <string>

(Read-Write) Sets and reads the port of the ECal noise tuner that is connected to the PNA SOURCE OUT.

Parameters

- <ch> Noise Figure channel number. If unspecified, value is set to 1.
- <string> ECal port identifier. Case sensitive.

Examples

```
SENS:NOIS:TUN:INP "B"  
sense:noise:tuner:input "A"
```

Query Syntax SENSE:NOISe:TUNer:INPut?

Return Type String

Default "B"

SENSe<ch>:NOISe:TUNer:ORlent[:STATe] <bool>

(Read-Write) Sets the state of auto orientation for a noise tuner during Noise Figure for NFX.

Parameters

- <ch> Noise Figure channel number. If unspecified, value is set to 1.
- <bool> Auto-orientation state. Choose from:
 - OFF** or **0** - Disable Auto-orientation
 - ON** or **1** - Enable Auto-orientation

Examples

```
SENS:NOIS:TUN:ORI 0  
sense2:noise:tuner:orient:state ON
```

Query Syntax SENSE:NOISE:TUNer:ORlent[:STATE]?

Return Type Boolean

Default 1 - ON

SENSe<ch>:NOISe:TUNer:OUTPut <string>

(Read-Write) Sets and reads the port of the ECal noise tuner that is connected to the CPLR THRU.

Parameters

- <ch> Noise Figure channel number. If unspecified, value is set to 1.
- <string> ECal port identifier. Case sensitive.

Examples

```
SENS:NOIS:TUN:OUTP "B"  
sense:noise:tuner:output "A"
```

Query Syntax SENSE:NOISE:TUNer:OUTput?

Return Type String

Default "A"

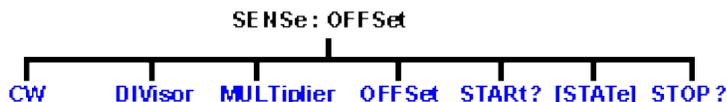
Last Modified:

- 21-Jun-2013 Added missing question mark to query
- 24-Oct-2012 Add snp save
- 9-Jun-2011 Added RMethod (A.09.41)
- 11-Jan-2011 Added count keyword to average factor
- 2-Mar-2010 Added Noise Receiver command (9.2)
- 6-Oct-2009 Added NFX commands (9.1)
- 21-Jun-2007 MX New topic

Sense:Offset Commands **Superseded**

Note: These commands are replaced by the [Sense:FOM](#) commands which include the features of the new FOM dialog. Although these old commands will continue to work, they can NOT be mixed with the new commands.

Sets the offset frequency functions, causing the stimulus and response frequencies to be different.



Click on a [blue](#) keyword to view the command details.

See Also

- [Example Programs](#)
- [Learn about Frequency Offset](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

SENSE<cnum>:OFFSet:CW <bool>

(Read-Write) Turns stimulus CW Override mode ON or OFF. Use this setting to establish a fixed (CW) stimulus frequency while measuring the Response over a swept frequency range.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<bool> ON (or 1) - turns CW override ON.

OFF (or 0) - turns CW override OFF.

Examples

```
SENSE:OFFS:CW ON
sense2:offset:cw off
```

Query Syntax SENSE<cnum>:OFFSet:CW?

Return Type Boolean

Default OFF

SENSE<cnum>:OFFSet:DIVisor <num>

(Read-Write) Specifies (along with the multiplier) the value to multiply by the stimulus.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <num> Divisor value. Range is 1 to 1000

Examples

```
SENS:OFFS:DIV 3  
sense2:offset:divisor 2
```

Query Syntax SENSE<cnum>:OFFSet:DIVisor?

Return Type Numeric

Default 1

SENSe<cnum>:OFFSet:MULTplier <num>

(Read-Write) Specifies (along with the divisor) the value to multiply by the stimulus.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <num> Multiplier value. Range is +/- 1000. Negative multipliers cause the stimulus to sweep in decreasing direction. For mixer measurements, this would be for setups requiring the RF frequency to be less than LO frequency

Examples

```
SENS:OFFS:MULT 2  
sense2:offset:multiplier 4
```

Query Syntax SENSe<cnum>:OFFSet:MULTplier?

Return Type Numeric

Default 1

SENSe<cnum>:OFFSet:OFFSet <num>

(Read-Write) Specifies an absolute offset frequency in Hz. For mixer measurements, this would be the LO frequency.

Parameters

<num> Any existing channel number. If unspecified, value is set to 1

<num> Offset frequency. Range is +/- 1000 GHz. Offsets can be positive or negative

Examples

```
SENS:OFFS:OFFS 1GHz  
sense2:offset:offset 1e9
```

Query Syntax SENSE<num>:OFFSet:OFFSet?

Return Type Numeric

Default 0 Hz

SENSe<num>:OFFSet:START?

(Read-Only) Returns the response start frequency

Parameters

<num> Any existing channel number. If unspecified, value is set to 1

Examples

```
SENS:OFFS:STAR?  
sense2:offset:start?
```

Return Type Numeric

Default Not applicable

SENSe<num>:OFFSet:[STATe] <bool>

(Read-Write) Enables Frequency Offset Mode on ALL measurements that are present on the active channel. This immediately causes the source and receiver to tune to separate frequencies. The receiver frequencies are specified with the other SENS:OFFSet commands. To make the stimulus settings use the [SENS:FREQ](#) commands.

Tip: To avoid unnecessary errors, first make other offset frequency settings, then set Frequency Offset ON.

Parameters

<num> Any existing channel number. If unspecified, value is set to 1

<bool> ON (or 1) - turns Frequency Offset ON.

OFF (or 0) - turns Frequency Offset OFF.

Examples

```
SENS:OFFS ON  
sense2:offset:state off
```

Query Syntax SENSE<cnum>:OFFSet:[STATe]?

Return Type Boolean

Default OFF (0)

SENSe<cnum>:OFFSet:STOP?

(Read-Only) Returns the response stop frequency.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

Examples

```
SENS:OFFS:STOP  
sense2:offset:stop
```

Return Type Numeric

Default Not applicable

Last Modified:

23-Jul-2007 Superseded Note.

Sense:Path:Config Commands

Controls the path configuration settings.

SENSe:PATH:CONFig

| [CATalog?](#)

| [COPY](#)

| [DELeTe](#)

| [DTEXT](#)

| [ELEMeNt](#)

 | [CATalog?](#)

 | [\[STATe\]](#)

 | [VALue:CATalog](#)

| [NAME](#)

| [SELeCt](#)

| [STORE](#)

Click on a [blue](#) keyword to view the command details.

The 'ELEMeNt' commands are used for both RF path and IF path configuration.

- [RF Configuration elements and values](#)
- [IF Configuration elements and values](#)

See Also

- [Example Programs](#)
- [Learn about RF Path Configuration](#)
- [Learn about IF Path Configuration](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

SENSe:PATH:CONFig:CATalog?

(Read-only) Returns a list of configuration names stored in the PNA.

Examples `SENS:PATH:CONF:CAT?`

Return Type Comma-separated list of double-quoted strings

Default Not Applicable

SENSe<ch>:PATH:CONFig:COpy <num>

(Write-only) Copies the mechanical switch and attenuator settings from the specified channel <num> to channel <ch>.

To avoid potential conflicts, all port couplings in the calling channel will be turned OFF and all port attenuator settings will be set to manual before copying the switch or attenuator settings. The two channels CAN be of different measurement classes.

Use [SYSTem:MACRo:COpy:CHANnel](#) to copy ALL settings from one channel to another.

Parameters

<ch> Channel number to copy mechanical settings to. If unspecified, value is set to 1.

<num> Channel number to copy mechanical settings from.

Examples `'Copies mechanical settings from chan 2 to chan 1.'`

`SENS1:PATH:CONF:COpy 2`

Return Type Not Applicable

Default Not Applicable

SENSe:PATH:CONFig:DELeTe <string>

(Write-only) Deletes the specified configuration name from the PNA. The factory configurations cannot be deleted. This is the only method of distinguishing a factory configuration from a user-named configuration.

Parameters

<string> Configuration name to be deleted.

Examples `SENS:PATH:CONF:DEL "MyMixer"`

Return Type Not Applicable

Default Not Applicable

SENSe:PATH:CONFig:DTExt <string>

(Read-Write) Write and read descriptive text associated with the configuration. This text is displayed in the path configuration dialog. Text is generally used to describe external connections that must be made manually to complete the configuration setup.

Parameters

<string> Descriptive text enclosed in quotes. Double quotes are not allowed within the descriptive text.

Examples

```
SENS:PATH:CONF:DTEX "Connect J1 jumper on the rear panel."
```

Query Syntax

SENSe<ch>:PATH:CONFig:DTEXt?

Return Type

String

Default

Not Applicable

SENSe<ch>:PATH:CONFig:ELEMent:CATalog?

(Read-only) Returns the names of configurable elements as a comma-delimited list of strings.

See a [list of configurable elements and settings](#) for various PNA models.

Parameters

<ch> Any existing channel number; if unspecified, value is set to 1.

Examples

```
SENS:PATH:CONF:ELEM:CAT?
```

```
'returns
```

```
"Combiner", "Src1", "Src2"
```

Default

Not Applicable

SENSe<ch>:PATH:CONFig:ELEMent[:STATe] <elem>, <setting>

(Read-Write) Write or read the setting of a specified element in the current configuration.

This command is used for both RF path and IF path configuration.

- [RF Configuration elements and values](#)
- [IF Configuration elements and values](#) (Includes IF Path, Pulse, and DSP elements)

Parameters

- <ch> Any existing channel number; if unspecified, value is set to 1.
- <elem> Name of the element for which a setting is to be made.
- <setting> Element setting. Use [SENS:PATH:CONF:ELEM:VAL:CAT?](#) to return a list of valid settings for the specified element.

Examples

```
SENS:PATH:CONF:ELEM "Combiner", "Normal"
```

Query Syntax

```
SENSe<ch>:PATH:CONFig:ELEMent? "Combiner"
```

Returns the current state of the Combiner element.

Return Type

String

Default

Not applicable

SENSe<ch>:PATH:CONFig:ELEMent:VALue:CATalog? <element>

(Read-only) Returns the list of valid settings that can be used with the specified element.

See a [list of configurable elements and settings](#) for various PNA models.

Parameters

- <ch> Any existing channel number; if unspecified, value is set to 1.
- <element> String. Element name for which to return valid settings.

Examples

```
SENS:PATH:CONF:ELEM:VAL:CAT? "Combiner"
```

```
'returns "Normal", "Reversed"
```

Default

Not Applicable

SENSe<ch>:PATH:CONFig:NAME?

(Read-only) Returns the name of the current configuration only if NO individual element settings had been changed since selecting or storing a configuration. When element settings change, the path configuration name is cleared.

Parameters

<ch> Any existing channel number; if unspecified, value is set to 1.

Examples

```
SENS:PATH:CONF:NAME?
```

```
'returns "Default"
```

Return Type String

Default Not Applicable

SENSe<ch>:PATH:CONFig:SElect <string>

(Write only) Loads the named configuration onto the specified channel.

Note: Loading a stored configuration will over-write MANY RF and [IF path configuration](#) settings. Make your measurement settings AFTER recalling a stored configuration, NOT before.

Use [SENS:PATH:CONF:CAT?](#) to return the configuration names that are stored on the PNA.

Parameters

<ch> Any existing channel number; if unspecified, value is set to 1.

<string> Configuration name. "Default" is the default factory configuration.

Examples

```
SENS:PATH:CONF:SEL 'default'
```

```
sense2:path:CONFig:select "MyMixer"
```

Query Syntax Not Applicable

Default "Default"

SENSe<ch>:PATH:CONFig:STORe <name>

(Write only) Saves the path configuration currently associated with channel <ch> to the specified configuration name.

Parameters

- <ch> Any existing channel number; if unspecified, value is set to 1.
- <name> String. Configuration name. Factory configurations can NOT be overwritten. Specifying the name of a pre-defined factory configuration will result in an error.

Examples

```
SENS:PATH:CONF:STOR "MyMixer"
```

Query Syntax Not Applicable

Default Not Applicable

Last Modified:

- 11-Jul-2013 Added note at State command
- 18-Oct-2012 Added Load note
- 27-Mar-2012 Minor edits for element, value
- 3-May-2011 Added copy command
- 24-Mar-2010 Changed CONFiguration to CONFig
- 14-Dec-6 MX New Topic

Sense:Power Command

[Learn about Receiver Attenuation](#)

SENSe<cnUm>:POWer:ATTenuator <recvr>,<num>

(Read-Write) Sets the attenuation level for the specified receiver.

Note: Attenuation cannot be set with Sweep Type set to Power

Parameters

<cnUm> Any existing channel number. If unspecified, value is set to 1

<recvr> Receiver to get attenuation. Choose from:

- **ARECeiver** - receiver A
- **BRECeiver** - receiver B
- **CRECeiver** - receiver C
- **DRECeiver** - receiver D

Receiver attenuation can NOT be set using [logical receiver notation](#).

<num> Attenuation value in dB. To determine how many receiver attenuators, the maximum receiver attenuation, and attenuation step size, for a PNA model, see [PNA Models and Options](#).

If a number other than these is entered, the analyzer will select the next lower valid value. For example, if 19 is entered for the E8361A/C, then 10 dB attenuation will be selected.

Examples

```
SENS:POW:ATT AREC,10  
sense2:power:attenuator breceiver,30
```

Query Syntax SENSe<cnUm>:POWer:ATTenuator? <recv>

Return Type Numeric

Default 0

Last Modified:

July 9, 2009 Edit syntax

25-Oct-2007 Edit range of values

Sense:Pulse Commands

Configures the 5 pulse generators in the PNA-X.

Beginning with A.09.50, these commands can also be used to control an external Pulse Generator.

[Learn more.](#)

SENSe:PULSe

- | [CATalog?](#)
- | [DELay](#)
- | [DINCrement](#)
- | [INVert](#)
- | [PERiod](#)
- | [STATe](#)
- | [SUBPointtrig](#)
- | [TPOLarity](#)
- | [TTYPe](#)
- | [WIDTh](#)

Click on a [blue](#) keyword to view the command details.

To make other Pulse settings, such as enabling the internal pulse modulators, use this command:

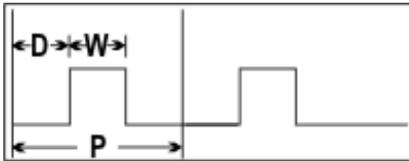
- [SENSe<ch>:PATH:CONFig:ELEMent\[:STATe\] <elem>, <setting>](#)
- At that command help topic, click the **IF Configuration elements and values** link to see the Pulse element and setting (in the middle box) to configure.

See Also

- [SENS:SWEep:PULSE](#) - configures the channel for pulse measurements
- [External Pulse Generator configuration commands](#)
- [SENS:IF configuration commands](#)
- [Example Programs](#)
- [PNA-X Integrated Pulse Application](#)

- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

Pulse Definitions



- D = Delay; the time before each pulse begins
- W = Width; the time the pulse is ON
- P = Period; one complete pulse cycle
- Duty Cycle = W/P

Important: If $D + W$ is greater than P , then undefined PNA behavior results. There is NO error message or warning.

SENSe:PULSe:CATalog?

(Read-only) Returns the string names of internal and configured external pulse generators.

Parameters None

Examples `SENS:PULS:CAT?`

Default Not Applicable

SENSe<ch>:PULSe<n>:DELay <value>[,<name>]

(Read-Write) Sets the pulse delay. The amount of time before a new pulse begins.

[See Pulse Definition diagram.](#)

Parameters

- <ch> Any existing channel number; if unspecified, value is set to 1.
- <n> Internal pulse generator number. Choose from 0 to 4.
0 is the generator that pulses the ADC.
- <value> Delay value in seconds. Choose a value from about 33ns to about 70 seconds.
- <name> Optional. String name of the pulse generator.
Required for use with [external pulse generators](#).
Use [SENSe:PULSe:CAT?](#) to return the names of configured pulse generators.
If specified, <n> is ignored.
If unspecified, <n> is required for internal pulse generators.

Examples

```
SENS:PULS1:DEL .5  
SENS:PULS:DEL .5, "My81110"
```

Query Syntax SENSE<ch>:PULSe<n>:DELay? [<name>]

Return Type Numeric

Default 0

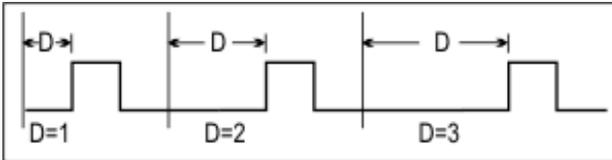
SENSe<ch>:PULSe<n>:DINCrement <value>[,<name>]

(Read-Write) Sets the pulse delay increment. The delay increments with each pulse by the <value> amount.

For example, in this diagram the delay starts as 1. On the second pulse, delay=2. On the third pulse, delay=3.

Important: If **D + W** is greater than **P**, then undefined PNA behavior results. There is NO error message or warning. Delay includes the incremented value.

This is useful for pulse profiling.



[See Pulse Definition diagram.](#)

Parameters

- <ch> Any existing channel number; if unspecified, value is set to 1.
- <n> Pulse generator number. Choose from 0 to 4.
0 is the generator that pulses the ADC.
- <value> Delay increment value in seconds.
- <name> Optional. String name of the pulse generator.
Required for use with [external pulse generators](#).
Use [SENSe:PULSe:CAT?](#) to return the names of configured pulse generators.
If specified, <n> is ignored.
If unspecified, <n> is required for internal pulse generators.

Examples

```
SENS:PULS1:DINC .5  
SENS:PULS:DINC .5, "My81110"
```

Query Syntax SENSE<ch>:PULSe<n>:DINCrement? [<name>]

Return Type Numeric

Default 0

SENSe<ch>:PULSe<n>:INVert <value>[,<name>]

(Read-Write) Sets whether to invert the polarity of the pulse.

Parameters

- <ch> Any existing channel number; if unspecified, value is set to 1.
- <n> Pulse generator number. Choose from 0 to 4.
0 is the generator that pulses the ADC.
- <value> Boolean
ON (or 1) - Invert the pulse generator polarity. This causes the pulse ON time to be active low and OFF be active high.
OFF (or 0) - Do NOT Invert the pulse generator polarity.
- <name> Optional. String name of the pulse generator.
Required for use with [external pulse generators](#).
Use [SENSe:PULSe:CAT?](#) to return the names of configured pulse generators.
If specified, <n> is ignored.
If unspecified, <n> is required for internal pulse generators.

Examples

```
SENS:PULS1:INV 1  
SENS:PULS:INV 1, "My81110"
```

Query Syntax SENSE<ch>:PULSe:INVert? [<name>]

Return Type Boolean

Default OFF (0)

SENSe<ch>:PULSe:PERiod <value>[,<name>]

(Read-Write) Sets the pulse-period (1/PRF) for ALL pulse generators.

The resolution of the period is:

DSP version: **4.0** = 16.667nS.

DSP version: **5.0** = 10nS

[Learn more](#) about DSP version.

[See Pulse Definition diagram.](#)

Parameters

- <ch> Any existing channel number; if unspecified, value is set to 1.
- <value> Pulse period in seconds. Choose a value from about 33ns to about 70 seconds.
- <name> Required for use with an [external pulse generator](#).
String name of the external pulse generator.
If unspecified, the period for the internal pulse generators are set.
Use [SENSe:PULSe:CAT?](#) to return the names of configured pulse generators.

Examples

```
SENSe:PULSe:PERiod .05  
SENSe:PULSe:PER .01, "My81110"
```

Query Syntax SENSE<ch>:PULSe:PERiod? [<name>]

Return Type Numeric

Default 1e-3 sec

SENSe<ch>:PULSe<n>[:STATe] <value>[,<name>]

(Read-Write) Turns the pulse output ON and OFF.

Parameters

- <ch> Any existing channel number; if unspecified, value is set to 1.
- <n> Pulse generator number. Choose from 0 to 4.
0 is the generator that pulses the ADC.
- <value> Boolean
ON (or 1) - turns pulse output ON.
OFF (or 0) - turns pulse output OFF.
- <name> Optional. String name of the pulse generator.
Required for use with [external pulse generators](#).
Use [SENSe:PULSe:CAT?](#) to return the names of configured pulse generators.
If specified, <n> is ignored.
If unspecified, <n> is required for internal pulse generators.

Examples

```
SENSe:PULS1 1  
SENSe:PULS 1, "My81110"
```

Query Syntax SENSE<ch>:PULSe[:STATe]? [<name>]

Return Type Boolean

Default OFF

SENSe<ch>:PULSe<n>:SUBPointtrig <bool>

(Read-Write) Enables / Disables subpoint triggering. When enabled and performing [Point Averaging](#), Each rising edge of P0 triggers a subpoint (one of N acquisitions in an N point average). Must also enable the P0 generator using [SENS:PULS0:STAT](#).

[Learn more about the PNA-X pulse generators.](#)

Parameters

- <ch> Any existing channel number; if unspecified, value is set to 1.
- <n> Pulse generator number. **Must be 0** as this is the generator that triggers the ADC.
- <bool> ON (or 1) - turns subpoint triggering ON.
OFF (or 0) - turns subpoint triggering OFF.

Examples

```
SENS:PULS0:SUBP 1
```

Query Syntax SENSE<ch>:PULSe0:SUBPointtrig?

Return Type Boolean

Default OFF

SENSe<ch>:PULSe:TPOLarity <char>

(Read-Write) Sets the polarity of the trigger signal to which the internal pulse generators will respond when being externally triggered at the PulseSyncIn pin.

Note: This feature requires DSP version: **4.0 FPGA: 34** or higher. [Learn more.](#)

[Learn more about the PNA-X pulse generators.](#)

Parameters

- <ch> Any existing channel number; if unspecified, value is set to 1.
- <char> Pulse polarity. Choose from:
POSitive - PNA responds to rising edge or HIGH level
NEGative - PNA responds to falling edge or LOW level.
Set Edge or Level triggering using [SENS:PULS:TTYPe](#).

Examples

```
SENS:PULS:TPOL NEG
```

Query Syntax SENSe<ch>:PULSe:TPOLarity?

Return Type Character

Default POSitive - Also the polarity used when the PNA-X does not have the required DSP hardware.

SENSe<ch>:PULSe<n>:TTYPe <char>

(Read-Write) Sets the type of trigger signal to which the internal pulse generators will respond when being externally triggered at the PulseSyncIn pin.

Note: This feature requires DSP version: **4.0 FPGA: 34** or higher. [Learn more.](#)

[Learn more about the PNA-X pulse generators.](#)

Parameters

- <ch> Any existing channel number; if unspecified, value is set to 1.
- <char> Trigger type. Choose from:
- EDGE** - PNA responds to the edge (rising or falling) of a signal
 - LEVeL** - PNA responds to the level (HIGH or LOW) of a signal
- Set polarity using [SENS:PULS:TPOL](#)

Examples

```
SENS:PULS:TTYP EDGE
```

Query Syntax SENSe<ch>:PULSe:TTYPe?

Return Type Character

Default LEVeL - Also the type used when the PNA-X does not have the required DSP hardware.

SENSe<ch>:PULSe<n>:WIDTh <value>[,<name>]

(Read-Write) Sets the pulse width. The amount of time that the pulse is ON.

[See Pulse Definition diagram.](#)

Parameters

- <ch> Any existing channel number; if unspecified, value is set to 1.
- <n> Pulse generator number. Choose from 0 to 4.
0 is the generator that pulses the ADC.
- <value> Pulse width in seconds. Choose a value from about 33ns to about 70 seconds.
- <name> Optional. String name of the pulse generator.
Required for use with [external pulse generators](#).
Use [SENSe:PULSe:CAT?](#) to return the names of configured pulse generators.
If specified, <n> is ignored.
If unspecified, <n> is required for internal pulse generators.

Examples

```
SENSe:PULSe:WIDTh .5  
SENSe:PULSe:WIDTh .5, "My81110"
```

Query Syntax SENSE<ch>:PULSe<n>:WIDTh? [<name>]

Return Type Numeric

Default 1e-4 sec

Last Modified:

19-Aug-2013	Added <name> to Period
16-Feb-2012	Added <name> for ext PG
11-Jan-2011	Minor edit
8-Oct-2010	Added Invert
9-Dec-2009	Added Trigger Polarity and Type (9.1.3)
20-Jul-2009	Added subpoint trig (8.55.09)
2-Jan-2007	MX New topic

Sense:Roscillator Command

[Learn about the Reference Osc.](#)

[See the rear-panel 10 MHz connector.](#)

SENSe:ROSCillator:SOURce?

(Read-only) Applying a 10 MHz signal to the Reference Oscillator connector automatically sets the Reference Oscillator to EXTERNAL. This command allows you to check that it worked.

- **EXT** is returned when a signal is present at the 10 MHz Reference Oscillator connector.
- **INT** is returned when NO signal is present at the 10 MHz Reference Oscillator connector.

Examples

```
SENS:ROSC:SOUR?  
sense:roscillator:source?
```

Return Type Character

Default Not applicable

Sense:Segment Commands

Defines the segment sweep settings.

Enable segment sweep with [SENS:SWE:TYPE SEGMENT](#).

SENSe:Segment

| **ADD**

| **ARbitrary**

| **BWIDth**

 | **[RESolution]**

 | **CONTRol**

| **COUNT**

| **DELeTe**

 | **ALL**

| **FREQuency**

 | **CENTer**

 | **SPAN**

 | **STARt**

 | **STOP**

| **LIST**

| **POWer**

 | **[LEVel]**

 | **CONTRol**

| **[STATe]**

| **SWEep**

 | **POINTs**

 | **TIME**

 | **CONTRol**

| **X**

Click on a [blue](#) keyword to view the command details.

See Also

- Example: [Upload and Download a Segment List](#)
- [Learn about Segment Sweep](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

SENSe<cnum>:SEGMENT<snum>:ADD

(Write-only) Adds a segment.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <snum> Segment number to add. If unspecified, value is set to 1. Segment numbers must be sequential. If a new number is added where one currently exists, the existing segment and those following are incremented by one.

Examples

Two Segments exist (1 and 2). The following command will add a new segment (1). The existing (1 and 2) will become (2 and 3) respectively.

```
SENS:SEGM1:ADD  
sense2:segment1:add
```

Query Syntax Not applicable. Use Sense:Segment:Count to determine the number of segments in a trace.

Default Not Applicable

SENSe<cnum>:SEGMENT:ARbitrary <ON | OFF>

(Read-Write) Enables you to setup a segment sweep with arbitrary frequencies. The start and stop frequencies of each segment can overlap other segments. Also, each segment can have a start frequency that is greater than its stop frequency which causes a reverse sweep over that segment. Learn more about [Arbitrary Segment Sweep](#).

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1.
- <ON | OFF> **ON** (or 1) - Allows the setup of arbitrary segment sweep.
OFF (or 0) - Prevents the setup of arbitrary segment sweep.

Examples

```
SENS:SEGM:ARB ON  
sense2:segment:arbitrary off
```

Query Syntax SENSE<cnum>:SEGMENT:ARbitrary?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

SENSe<cnum>:SEGMENT<snum>:BWIDth[:RESolution] <num>

(Read-Write) Sets the IF Bandwidth for the specified segment. First set [SENS:SEGM:BWIDth:CONTRol](#) ON. All subsequent segments that are added assume the new IF Bandwidth value.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <snum> Segment number to modify. Choose any existing segment number.
- <num> IF Bandwidth in Hz. The list of valid IF Bandwidths is different depending on the PNA model. [\(Click to see the lists.\)](#) If an invalid number is specified, the analyzer will round up to the closest valid number.

Note: This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
SENS:SEGM:BWID 1KHZ  
sense2:segment2:bwid:resolution max
```

Query Syntax SENSE<cnum>:SEGMENT<snum>:BWIDth[:RESolution]?

Return Type Numeric

Default Varies with PNA model.

SENSe<cnum>:SEGMENT:BWIDth[:RESolution]:CONTRol <ON | OFF>

(Read-Write) Specifies whether the IF Bandwidth resolution can be set independently for each segment.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<ON | OFF> **ON** (or 1) - turns Bandwidth control ON. Bandwidth can be independently set for each segment.

OFF (or 0) - turns Bandwidth control OFF. Use the channel IF bandwidth setting [SENS:BWID](#).

Examples

```
SENS:SEGM:BWID:CONT ON
```

```
sense2:segment:bandwidth:control off
```

Query Syntax SENSe<cnum>:SEGMENT:BWIDth[:RESolution]:CONTRol?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

SENSe<cnum>:SEGMENT:COUNt?

(Read-only) Queries the number of segments that exist in the specified channel.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

Examples

```
SENS:SEGM:COUNt?
```

```
sense2:segment:count?
```

Return Type Numeric

Default 1 segment

SENSe<cnum>:SEGMENT<snum>:DELeTe

(Write-only) Deletes the specified sweep segment. When ALL segments are deleted, [SENS:SWE:TYPE](#) is automatically set to Linear because there are no segments to sweep.

<cnum> Any existing channel number. If unspecified, value is set to 1

<snum> Number of the segment to delete. If unspecified, value is set to 1

Examples

```
SENS:SEGM:DEL  
sense2:segment2:delete
```

Query Syntax Not applicable

Default Not Applicable

SENSe<cnum>:SEGMENT:DELeTE:ALL

(Write-only) Deletes all sweep segments. When this command is executed, [SENS:SWE:TYPE](#) is automatically set to Linear because there are no segments to sweep.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

Examples

```
SENS:SEGM:DEL:ALL  
sense2:segment:delete:all
```

Query Syntax Not applicable

Default Not Applicable

SENSe<cnum>:SEGMENT<snum>:FREQUency:CENTer <num>

(Read-Write) Sets the Center Frequency for the specified segment. The Frequency Span of the segment remains the same. The Start and Stop Frequencies change accordingly.

Note: All previous segment's Start and Stop Frequencies that are larger than the new Start Frequency are changed to the new Start Frequency. All following segment's start and stop frequencies that are smaller than the new Stop Frequency are changed to the new Stop Frequency.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<snum> Segment number to modify. Choose any existing segment number.

<num> Center Frequency in Hz. Choose any number between the **minimum** and **maximum** frequency of the analyzer.

Note: This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
SENS:SEGM:FREQ:CENT 1MHZ
sense2:segment2:frequency:center 1e9
```

Query Syntax SENSE<cnum>:SEGMent<snum>:FREQUency:CENTer?

Return Type Numeric

Default Stop Frequency of the previous segment. If first segment, start frequency of the analyzer.

SENSe<cnum>:SEGMent<snum>:FREQUency:SPAN <num>

(Read-Write) Sets the Frequency Span for the specified segment. The center frequency of the segment remains the same. The start and stop frequencies change accordingly.

Note: All previous segment's Start and Stop Frequencies that are larger than the new Start Frequency are changed to the new Start Frequency. All following segment's start and stop frequencies that are smaller than the new Stop Frequency are changed to the new Stop Frequency.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <snum> Segment number to modify. Choose any existing segment number.
- <num> Frequency Span in Hz. Choose any number between the **minimum** and **maximum** frequency of the analyzer.

Note: This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
SENS:SEGM:FREQ:SPAN 1MHZ
sense2:segment2:frequency:span max
```

Query Syntax SENSE<cnum>:SEGMent<snum>:FREQUency:SPAN?

Return Type Numeric

Default If first segment, frequency span of the analyzer. Otherwise 0.

SENSe<cnum>:SEGMent<snum>:FREQUency:START <num>

(Read-Write) Sets the Start Frequency for the specified sweep segment.

Notes

All other segment Start and Stop Frequency values that are larger than this frequency are changed to this frequency.

To return the start and stop frequency of the entire sweep (all segments), Use [SENS:FREQ:START?](#) and [SENS:FREQ:STOP?](#)

Parameters

- <num> Any existing channel number. If unspecified, value is set to 1
- <snum> Segment number to modify. Choose any existing segment number.
- <num> Start Frequency in Hz. Choose any number between the **minimum** and **maximum** frequency of the analyzer.

Note: This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
SENS:SEGM:FREQ:STAR 1MHZ  
sense2:segment2:frequency:start minimum
```

Query Syntax SENSE<num>:SEGMent<snum>:FREQuency:STARt?

Return Type Numeric

Default Stop Frequency of the previous segment. If first segment, start frequency of the analyzer.

SENSe<num>:SEGMent<snum>:FREQuency:STOP <num>

(Read-Write) Sets the Stop Frequency for the specified sweep segment.

Notes

All other segment Start and Stop Frequency values that are larger than this frequency are changed to this frequency.

To return the start and stop frequency of the entire sweep (all segments), Use [SENS:FREQ:START?](#) and [SENS:FREQ:STOP?](#)

Parameters

- <num> Any existing channel number. If unspecified, value is set to 1
- <snum> Segment number to modify. Choose any existing segment number.
- <num> Stop Frequency in Hz. Choose any number between the **minimum** and **maximum** frequency of the analyzer.

Note: This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
SENS:SEGM:FREQ:STOP 1MHZ
sense2:segment2:frequency:stop maximum
```

Query Syntax SENSE<cnum>:SEGMENT<snum>:FREQUENCY:STOP?

Return Type Numeric

Default If first segment, stop frequency of the analyzer. Otherwise, start frequency of the segment.

SENSe<cnum>:SEGMENT:LIST <char>,<numSegs>,<data>

(Read-Write) Reads or writes the entire list of values in the segment sweep table.

Note: For binary data transfer, specify 64-bit instead of 32-bit using [FORMAt\[:DATA\]](#). This is because higher frequencies used on PNA exceed the maximum value that can be represented by a 32-bit floating point number.

When sending/receiving this data as binary (FORMAt[:DATA] REAL,64), use [FORMAt:BORDER](#) to specify the correct 'endianness' (byte ordering) corresponding to your programming environment / computer platform.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1

<char> Choose from:

SSTOP - Frequency values are Start and Stop for each segment.

CSPAN - Frequency values are Center and Span for each segment.

<numSegs> Total number of sweep segments being input. This allows the PNA to determine how many values per-each-segment are in the input <data> block.

<data> A 2-dimensional array of Segment data as a single data block. The data elements within a segment are each represented as real floating-point numbers as follows, and the data block is formed by interleaving all the segments together consecutively:

1. Segment state (Boolean 1 for ON and 0 for OFF)
2. Number of Points in the segment
3. Start Freq (when <char> is SSTOP), or Center Freq (when <char> is CSPAN)
4. Stop Freq (when <char> is SSTOP), or Freq Span (when <char> is CSPAN)
5. IFBW (optional for the Write)
6. Dwell Time (optional for the Write)
7. Power (optional for the Write) - see below.

The first four data elements must always be supplied. After those values, data must be supplied for successive optional elements. For example, to set dwell time values, you must also supply IFBW values, because IFBW (#5) precedes dwell time (#6) in the array order.

The [IF Bandwidth](#), [Sweep Time](#) and [Source Power](#) Control settings do NOT affect the order in which elements are interpreted.

The number of elements to supply for Power depends on the following two settings:

1. [Source Power Option](#) - ON allows segments to have independent power levels.
2. [Couple Ports](#) = Off allows different power levels for each test port.

CouplePorts	SourcePowerOption	Number of Elements
False	False	Each port has its own channel-wide power setting, which is set using SOURCE:POWER[:LEVEL] . Provide exactly 7 elements per segment. The last element (power) is ignored.
False	True	Provide 6 elements + total number of ports. The first 7 elements are still interpreted the same. The remaining elements (in-order) are interpreted as the power levels to set on that segment for Ports 2 through N, where N is the total number of ports currently enabled for the PNA or for a PNA with multiport external test set.
True	False	Provide exactly 7 elements per segment. The last element (power) is ignored.
True	True	Provide exactly 7 elements per segment. The last element (power) is honored.

Examples

```
SENS:SEGM:LIST SSTOP,1,1,201,10E6,26.5E9,1E3,0,-10 1 segment,
state ON, 201 points, 10 MHz to 26.5 GHz, 1kHz IFBW, 0 dwell
time, -10 dBm (port powers coupled)
```

```
sense2:segment:list? cspan
```

See [Upload and Download a Segment List](#) example program

Query Syntax SENSE<cnum>:SEGMENT:LIST? [char].

If unspecified, char is set to SSTOP.

The number of data elements per segment returned will be 6 + total number of source ports, regardless of the [IF Bandwidth](#), [Sweep Time](#) and [Source Power Control](#) settings. For the N5264A, which has no source ports, the query will return just 6 values per segment. For all other PNA models, the last elements in each segment correspond to the power level for each port.

Return Type Returns block data in the format specified by [FORMat\[:DATA\]](#).

Default Not Applicable

SENSe<cnum>:SEGMENT<snum>:POWER[<port>][:LEVel] <num>

(Read-Write) Sets the Port Power level for the specified sweep segment. First set SENS:SEGM:POW:CONTRol ON.

When [port power is Coupled](#), setting port power for one port will apply port power for all source ports.

All subsequent segments that are added assume the new Power Level value.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<snum> Segment number to modify. Choose any existing segment number.

<port> Port number of the source. If unspecified, value is set to 1.

<num> Power level.

Note: The range of settable power values depends on the PNA model and if source attenuators are installed. To determine the range of values, send SOUR:POW? MAX and SOUR:POW? MIN. ([SOUR:POW:ATT:AUTO](#) must be set to ON).

Actual achievable leveled power depends on frequency.

Examples

```
SENS:SEGM:POW 0
sense2:segment2:power1:level -10
```

Query Syntax SENSe<cnum>:SEGMENT<snum>:POWER[<port>][:LEVel]?

Return Type Numeric

Default 0

SENSe<cnum>:SEGMENT:POWER[:LEVel]:CONTRol <ON | OFF>

(Read-Write) Specifies whether Power Level can be set independently for each segment.

Parameters

- <cnun> Any existing channel number. If unspecified, value is set to 1
- <ON | OFF> **ON** (or 1) - turns Power Level control ON. Power level can be set for each segment.
OFF (or 0) - turns Power Level control OFF. Use the channel power level setting.

Examples

```
SENS:SEGM:POW:CONT ON  
sense2:segment:power:level:control off
```

Query Syntax SENSE<cnun>:SEGMENT:POWER[:LEVEL]:CONTROL?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

SENSE<cnun>:SEGMENT<snum>[:STATE] <ON | OFF>

(Read-Write) Turns the specified sweep segment ON or OFF. At least ONE segment must be ON or [Sweep Mode](#) is automatically set to **Linear**.

Parameters

- <cnun> Any existing channel number. If unspecified, value is set to 1
- <snum> Segment number to be turned ON or OFF
- <ON | OFF> **ON** (or 1) - turns segment ON.
OFF (or 0) - turns segment OFF.

Examples

```
SENS:SEGM ON  
sense2:segment2:state off
```

Query Syntax SENSE<cnun>:SEGMENT<snum>[:STATE]?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

SENSE<cnun>:SEGMENT<snum>:SWEep:POINTs <num>

(Read-Write) Sets the number of data points for the specified sweep segment.

Parameters

- <cnun> Any existing channel number. If unspecified, value is set to 1
- <snum> Any existing segment number. If unspecified, value is set to 1
- <num> Number of points in the segment. The total number of points in all segments cannot exceed **20001**. A segment can have as few as 1 point.

Note: This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
SENS:SEGM:SWE:POIN 51  
sense2:segment2:sweep:points maximum
```

Query Syntax SENSE<cnun>:SEGMent<snum>:SWEep:POINTs?

Return Type Numeric

Default 21

SENSE<cnun>:SEGMent<snum>:SWEep:TIME <num>

(Read-Write) Sets the time the analyzer takes to sweep the specified sweep segment.

Parameters

- <cnun> Any existing channel number. If unspecified, value is set to 1
- <snum> Any existing segment number.
- <num> Sweep time in seconds. Choose a number between **0** and **100**

Note: This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
SENS:SEGM:SWE:TIME 1ms  
sense2:segment2:sweep:time .001
```

Query Syntax SENSE<cnun>:SEGMent<snum>:SWEep:TIME?

Return Type Numeric

Default Not Applicable

SENSE<cnun>:SEGMent:SWEep:TIME:CONTrol <ON | OFF>

(Read-Write) Specifies whether Sweep Time can be set independently for each sweep segment.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<ON | OFF> **ON** (or 1) - turns Sweep Time control ON. Sweep Time can be set for each segment.

OFF (or 0) - turns Sweep Time control OFF. Uses the channel Sweep Time setting.

Examples

```
SENS:SEGM:SWE:TIME:CONT ON
sense2:segment:sweep:time:control off
```

Query Syntax SENSE<cnum>:SEGMENT:SWEep:TIME:CONTROL?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

SENSE<cnum>:SEGMENT:X:SPACING <char>

(Read-Write) Sets X-axis spacing ON or OFF

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<char> **LINear** - turns X-axis point spacing OFF

OBASe - turns X-axis point spacing ON

Examples

```
SENS:SEGM:X:SPACING LIN
sense2:segment:spacing obase
```

Query Syntax SENSE<cnum>:SEGMENT:X:SPACING?

Return Type Character

Default LINear

Last Modified:

- 20-Aug-2013 Fixed error in finder map.
- 7-May-2012 Removed preset
- 30-Aug-2011 Removed Snum from X:Spacing
- 27-Oct-2009 Added Segment:List note
- 29-Apr-2009 Added List command (8.6)
- 13-May-2008 Fixed segment delete links
- 21-Jun-2007 Increased max number of points

Sense:Sweep Commands

Specifies the sweep functions of the analyzer.

SENSe:SWEEp:

DWELI

| **AUTO**

| **SDELay**

GENeration

| **POINTsweep**

GROups

| **COUNT**

MODE

POINTs

PULSe More commands

SPEed

SRCPort

TIME

| **AUTO**

TRIGger

| **DELAY**

| **MODE**

| **POINT**

TYPE

| **FACW**

Click on a [blue](#) keyword to view the command details.

See Also

- [Example Programs](#)

- **Example** [Triggering the PNA using SCPI](#)
 - [Learn about Sweeping](#)
 - [Synchronizing the PNA and Controller](#)
 - [SCPI Command Tree](#)
-

SENSe<cnum>:SWEep:DWELI <num>

(Read-Write) Sets the dwell time between each sweep point.

- Dwell time is **ONLY** available with SENSE:SWEep:GENeration set to **STEPped**; It is **Not** available in **ANALOG**.
- Sending dwell = 0 is the same as setting SENS:SWE:DWEL:AUTO **ON**. Sending a dwell time > 0 sets SENS:SWE:DWEL:AUTO **OFF**.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<num> Dwell time in seconds.

This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
SENS:SWE:DWEL .1
sense2:sweep:dwell min
```

Query Syntax SENSe<cnum>:SWEep:DWELI?

Return Type Numeric

Default 0 - (**Note:** dwell time set to 0 is the same as dwell:auto ON)

SENSe<cnum>:SWEep:DWELI:AUTO <ON | OFF>

(Read-Write) Specifies whether or not to automatically calculate and set the minimum possible dwell time. Setting Auto **ON** has the same effect as setting dwell time to **0**.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<ON | OFF> **ON** (or 1) - turns dwell ON.
OFF (or 0) - turns dwell OFF.

Examples

```
SENS:SWE:DWEL:AUTO ON  
sense2:sweep:dwell:auto off
```

Query Syntax SENSE<cnum>:SWEep:DWELI:AUTO?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

SENSe<cnum>:SWEep:DWELI:SDELay <num>

(Read-Write) Specifies the time to wait just before acquisition begins for each sweep. This delay is in addition to [Dwell Time](#) and the following two External Trigger delays if enabled.

- [Trig:Delay](#) (global scope)
- [Sens:Swe:Trig:Delay](#) (channel scope)

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<num> Sweep delay in seconds.

This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
SENS:SWE:DWEL:SDEL .1  
sense2:sweep:dwell:sdelay .5
```

Query Syntax SENSE<cnum>:SWEep:DWELI:SDELay?

Return Type Numeric

Default 0

SENSe<cnum>:SWEep:GENeration <char>

(Read-Write) Sets sweep as Stepped or Analog.

Parameters

- <num> Any existing channel number. If unspecified, value is set to 1
- <char> Choose from:

STEPped - source frequency is CONSTANT during measurement of each displayed point. More accurate than ANALog. Dwell time can be set in this mode.

ANALog - source frequency is continuously RAMPING during measurement of each displayed point. Faster than STEPped. Sweep time (not dwell time) can be set in this mode.

Examples

```
SENS:SWE:GEN STEP  
sense2:sweep:generation analog
```

Query Syntax SENSE<num>:SWEep:GENeration?

Return Type Character

Default Analog

SENSe<num>:SWEep:GENeration:POINTsweep <bool>

(Read-Write) Turns ON and OFF point sweep mode. When enabled, the PNA measures both the forward and reverse parameters at each frequency point before stepping to the next frequency. [Learn more.](#)

Parameters

- <num> Any existing channel number. If unspecified, value is set to 1
- <char> Choose from:

ON or **(1)** - Enable point sweep mode.

OFF or **(0)** - Disable point sweep mode.

Examples

```
SENS:SWE:GEN:POIN 1  
sense2:sweep:generation:pointsweep off
```

Query Syntax SENSE<num>:SWEep:GENeration:POINTsweep?

Return Type Boolean

Default OFF

SENSe<cnum>:SWEep:GROups:COUNT <num>

(Read-Write) Sets the trigger count (groups) for the specified channel. Set trigger mode to group after setting this count.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <num> Count (groups) number. Choose any number between: **1** and **2e6** (1 is the same as single trigger)

Examples

```
SENS:SWE:GRO:COUN 10  
sense2:sweep:groups:count 50
```

Query Syntax SENSe<cnum>:SWEep:GROups:COUNT?

Return Type Numeric

Default 1

SENSe<cnum>:SWEep:MODE <char>

(Read-Write) Sets the number of trigger signals the specified channel will ACCEPT.

See [Triggering the PNA Using SCPI](#).

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <char> Trigger mode. Choose from:

HOLD - channel will not trigger

CONTinuous - channel triggers indefinitely

GROups - channel accepts the number of triggers specified with the last [SENS:SWE:GRO:COUN](#) <num>. This is one of the PNA overlapped commands. [Learn more](#).

SINGLE - channel accepts ONE trigger, then goes to HOLD.

Note: To perform simple, single-triggering, use SINGLE which requires that [TRIG:SOURce](#) remain in the default (internal) setting.

Examples

```
SENS:SWE:MODE CONT  
sense2:sweep:mode hold
```

Query Syntax SENSe<cnum>:SWEep:MODE?

Return Type Character

SENSe<cnum>:SWEep:POINts <num>

(Read-Write) Sets the number of data points for the measurement.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<num> Choose any number between **1** and the [PNA maximum number of points](#).

This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

Examples

```
SENS:SWE:POIN 51
sense2:sweep:points max
```

Query Syntax SENSe<cnum>:SWEep:POINts?

Return Type Numeric

Default 201

SENSe<cnum>:SWEep:SRCPort <1 | 2> **Superseded**

This command is superseded. The [Calc:Par:Def:Ext](#) and [Calc:Par:Mod:Ext](#) can now optionally include the source port.

(Read-Write) Sets the source port when making non S-parameter measurements. Has no effect on S-parameter measurements.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<1 | 2> **1** - Source power comes out Port 1

2 - Source power comes out Port 2

Examples

```
SENS:SWE:SRCP 1
sense2:sweep:srcport 2
```

Query Syntax SENSe<cnum>:SWEep:SRCPort?

Return Type Character

Default 1

SENSe<cnum>:SWEep:SPEed <char>

(Read-Write) Sets and returns the state of Fast Sweep mode. [Learn more about Fast Sweep.](#)

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<char> Fast Sweep mode. Choose from:

FAST - turns Fast Sweep Mode ON

NORMAL - turns Fast Sweep Mode OFF (Normal Mode).

Examples

```
SENS:SWE:SPE NORM
sense2:sweep:speed fast
```

Query Syntax SENSe<cnum>:SWEep:SPEed?

Return Type Character

Default NORMAL

SENSe<cnum>:SWEep:TIME <num>

(Read-Write) Sets the time the analyzer takes to complete one sweep. If sweep time accuracy is critical, use ONLY the values that are attained using the up and down arrows next to the sweep time entry box. [See Sweep Time.](#)

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<num> Sweep time in seconds.

To select the fastest sweep speed, either send MIN as an argument to this command, or send SENS:SWE:TIME:AUTO 1.

This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

The MAX value will change based on point count, IFBW, and dwell time.

Examples

```
SENS:SWE:TIME 1ms
sense2:sweep:time .001
```

Query Syntax SENSe<cnum>:SWEep:TIME?

Return Type Numeric

Default NA

SENSe<cnum>:SWEep:TIME:AUTO <ON | OFF>

(Read-Write) Turns the automatic sweep time function ON or OFF.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<ON | OFF> **ON** (or 1) - turns the automatic sweep time ON.

OFF (or 0) - turns the automatic sweep time OFF.

Examples

```
SENS:SWE:TIME:AUTO
sense2:sweep:time:auto off
```

Query Syntax SENSe<cnum>:SWEep:TIME:AUTO?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

SENSe<cnum>:SWEep:TRIGger:DELay <num>

(Read-Write) Sets and reads the trigger delay for all measurements in the specified CHANNEL. This delay is only applied while [TRIG:SOURce EXTERNAL](#) and [TRIG:SCOP CURRENT](#) . After an external trigger is applied, the start of the sweep is delayed for the specified delay value plus any inherent latency.

To apply a trigger delay for all channels (Global), use [TRIG:DEL](#)

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<num> Trigger delay value in seconds. Range is from 0 to 107 seconds

Examples

```
SENS:SWE:TRIG:DELay .003
sense2:sweep:trigger:delay 1
```

Query Syntax SENSe<cnum>:SWEep:TRIGger:DELay?

Return Type Numeric

Default 0

SENSe<cnum>:SWEep:TRIGger:MODE <char>

(Read-Write) Sets and reads the trigger mode for the specified channel. This determines what EACH signal will trigger. [Learn more.](#)

Note: Setting Point and Sweep mode forces [Trigger:SCOPE](#) = CURRENT

Parameters

<num> Any existing channel number. If unspecified, value is set to 1

<char> Trigger mode. choose from:

- **CHANnel** - Each trigger signal causes **ALL traces** in that channel to be swept.
- **SWEep** - Each Manual or External trigger signal causes **ALL traces that share a source port** to be swept.
- **POINT** -- Each Manual or External trigger signal causes one data point to be measured.
- **TRACe** - Allowed ONLY when [SENS:SWE:GEN:POIN](#) is enabled. Each trigger signal causes two identical measurements to be triggered separately - one trigger signal is required for each measurement. Other trigger mode settings cause two identical parameters to be measured simultaneously.

Examples

```
SENS:SWE:TRIG:MODE SWE  
sense2:sweep:trigger:mode point
```

Query Syntax SENSE<num>:SWEep:TRIGger:MODE?

Return Type Character

Default Channel

SENSe<num>:SWEep:TRIGger:POINT <ON | OFF> **Superseded**

This command is replaced with [SENS:SWE:TRIG:MODE POINT](#)

(Read-Write) Specifies whether the specified channel will measure one point for each trigger or all of the measurements in the channel. Setting any channel to POINT mode will automatically set the [TRIGger:SCOPE](#) = CURRent.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<ON | OFF> **ON** (or 1) - Channel measures one data point per trigger.

OFF (or 0) - All measurements in the channel made per trigger.

Examples

```
SENS:SWE:TRIG:POIN ON  
sense2:sweep:trigger:point off
```

Query Syntax SENSE<cnum>:SWEep:TRIGger:POINT?

Return Type Boolean (1 = Point, 0 = Measurement)

Default 0 - Measurement

SENSe<cnum>:SWEep:TYPE <char>

(Read-Write) Sets the type of analyzer sweep mode. First set sweep type, then set sweep parameters such as frequency or power settings.

Note: For SweptIMD channels, use [SENS:IMD:SWE:TYPE SEGMENT](#)

See Also: [FCA Segment Sweep commands](#)

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1

<char> Choose from:

LINear | LOGarithmic | POWer | CW | SEGMENT | PHASe

Note: SWEep TYPE cannot be set to SEGMENT if there are no segments turned ON. A segment is automatically turned ON when the analyzer is started.

Examples

```
SENS:SWE:TYPE LIN  
sense2:sweep:type segment
```

Query Syntax SENSe<cnum>:SWEep:TYPE?

Return Type Character

Default LINear

SENSe<cnum>:SWEep:TYPE:FACW <num>

(Read-Write) Enables Fast CW sweep and sets the number of data points for the channel. [Sweep Type](#) must already be set to CW and FIFO must already be enabled.

See Also

[FIFO commands](#)

[Example program](#)

[N5264A Measurement Receiver](#)

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <num> Number of data points to measure in Fast CW mode. This setting overwrites the standard number of points setting for the channel.

Set to 0 to disable Fast CW.

Examples

```
SENS:SWE:TYPE:FACW 1e6  
sense2:sweep:type facw 1e3
```

Query Syntax SENSe<cnum>:SWEep:TYPE:FACW?

Return Type Numeric

Default 0 - Disabled

Last Modified:

5-Mar-2014	Edits to sweep time max.
27-Feb-2014	Edits to Trig mode
8-Mar-2013	Added notes to sweep:type command
6-Mar-2012	Edited links at SDElay
3-Dec-2010	Added sweep type Phase
6-May-2010	Added sweep type note
1-Mar-2010	Added Sweep Delay
10-Oct-2008	Added FACW and point sweep
26-Aug-2008	Added sweep speed
21-Jun-2007	Increased max number of points
18-Jun-2007	Added Single to Mode
24-Apr-2007	Clarified Sweep mode

X Values Command

SENSe<cnum>:X[:VALues]? - **Superseded**

Replaced with [CALC:X?](#)

(Read-only) Returns the stimulus values for the specified channel. If the channel is sweeping the source backwards, the values will be in descending order.

Note: To avoid frequency rounding errors, specify [FORM:DATA](#) <Real,64> or <ASCIi, 0>

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1.

Examples

```
SENS:X?  
sense2:x:values?
```

Return Type Depends on [FORM:DATA](#) command

Default Not applicable

Last Modified:

16-Aug-2013	Superseded per JE
10-Sep-2012	Added Form:Data note
9-Apr-2010	Added Note

Source Commands

Controls the power delivered to the DUT.

SOURce:

[CATalog?](#)

[DC - More commands](#)

[PHASe - More commands](#)

POWer

| [ALC:MODE](#)

| [CATalog?](#)

| [RECeiver - More commands](#)

| [ATTenuation](#)

| [AUTO](#)

| [CENTer](#)

| [CORRection - More commands](#)

| [COUPle](#)

| [DETeCTOR](#)

| [\[LEVel\]](#)

| [\[IMMEDIATE\]\[AMPLitude\]](#)

| [SLOPe](#)

| [STATe](#)

| [MODE](#)

| [PORT](#)

| [STARt](#)

| [STOP](#)

| [SPAN](#)

| [STARt](#)

| [STOP](#)

Click on a [blue](#) keyword to view the command details.

See Also

- [Example Programs](#)
 - [Learn about Power Settings](#)
 - [Synchronizing the PNA and Controller](#)
 - [SCPI Command Tree](#)
 - [Remotely Specifying a Source Port](#)
-

SOURce<cnum>:CATalog?

(Read-only) Returns a list of valid port names that can be controlled. Some ports only have string names, NOT numbers. All commands that require a port argument have provisions for specifying either a port number OR a string name.

See also: [Remotely Specifying a Source Port](#).

Parameters

Examples

```
SOUR:CAT?  
source:catalog  
  
'Some PNA-X models return  
"Port 1,Port 2,Port 3,Port 4,Port 1 Src2"'
```

Return Type Comma-separated list of strings.

Default Not applicable

SOURce<cnum>:POWER<port>:ALC[:MODE] <char>, [src]

(Read-Write) Sets and returns the ALC mode for the specified channel and port. Use [SOUR:POW:ALC:MODE:CAT?](#) to return a list of valid ALC modes for the PNA.

[Learn more about ALC mode.](#)

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <port> Source port number of the PNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- <char> ALC Mode.

For the PNA-X choose from:

- **INTERNAL** Standard ALC loop
- **OPENloop** No ALC loop

[src] **String**. (NOT case sensitive). Source port. Optional. Use [SOUR:CAT?](#) to return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an [external source](#), [true mode balanced port](#), or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

```
SOUR:POW:ALC INT  
  
source2:power2:alc:mode openloop  
  
source:power:alc:mode openloop,"Port 1 Src2"
```

Query Syntax SOURce<cnum>:POWER<port>:ALC:MODE? [src]

Return Type Character

Default INTernal

SOURce<cnum>:POWER<port>:ALC[:MODE]:CATalog? [src]

(Read-only) Returns a list of valid ALC modes for the specified channel and port number. Use the returned values to set [SOUR:POW:ALC:MODE](#).

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <port> Source port number of the PNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- [src] **String**. (NOT case sensitive). Source port. Optional. Use [SOUR:CAT?](#) to return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an [external source](#), [true mode balanced port](#), or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

```
SOUR:POW:ALC:CAT?  
  
source2:power2:alc:mode:catalog?  
  
source:power:alc:mode:catalog? "Port 1 Src2"
```

Return Type Comma-separated list of strings.

Default Not applicable

SOURce<cnum>:POWER<port>:ATTenuation <num>, [src]

(Read-Write) Sets the attenuation level for the selected channel. Sending this command turns automatic attenuation control (SOUR:POW:ATT:AUTO) to OFF. If the ports are coupled, changing the attenuation on one port will also change the attenuation on all other ports. To turn port coupling OFF use [SOURce:POWER:COUPLE OFF](#).

Note: Attenuation cannot be set with **Sweep Type** set to **Power**

See [Sens:Power:ATT](#) to change receiver attenuation.

Parameters

- <num> Any existing channel number. If unspecified, value is set to 1
- <port> Source port number of the PNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- <num> Attenuation value. The range of settable values depends on the PNA model. To determine the valid settings, do one of the following:
- See [PNA models and options](#) to see the range and step size for each model / option.
 - Perform a query using MAX, then MIN, as an argument. Example:
`SOURce:POWer:ATT? Max` However, this will not tell you the attenuation step size.

If an invalid attenuation setting is entered, the PNA will select the next lower valid value. For example, if 19 is entered, then for an E8361A, 10 dB attenuation will be selected.

Note: This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

[src] **String.** (NOT case sensitive). Source port. Optional. Use [SOUR:CAT?](#) to return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an [external source](#), [true mode balanced port](#), or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

```
SOUR:POW:ATT 10  
  
source2:power2:attenuation maximum  
  
source:power:att 20, "Port 1 Src2"
```

Query Syntax `SOURce<num>:POWer<port>:ATTenuation? [min/max] [src]
[min/max,src]`

Return Type Numeric

Default 0

SOURce<cnum>:POWER<port>:ATTenuation:AUTO <bool>, [src]

(Read-Write) Turns automatic attenuation control ON or OFF. Setting an attenuation value (using SOURce:POWER:ATTenuation <num>) sets AUTO **OFF**.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1.
- <port> Source port number of the PNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- <bool> **ON** (or 1) - turns coupling ON. The analyzer automatically selects the appropriate attenuation level to meet the specified power level.
- OFF** (or 0) - turns coupling OFF. Attenuation level must be set using SOURce:POWER:ATTenuation <num>.
- [src] **String**. (NOT case sensitive). Source port. Optional. Use [SOUR:CAT?](#) to return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an [external source](#), [true mode balanced port](#), or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

```
SOUR:POW2:ATT:Auto On
source2:power:attenuation:auto off
sour:pow:att:auto 1, "Port 1 Src2"
```

Query Syntax SOURce<cnum>:POWER:ATTenuation:Auto? [src]

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

SOURce<cnum>:POWER<port>:CENTer <num>

(Read-Write) Sets the power sweep center power. Must also set: [SENS:SWE:TYPE POWER](#) and [SOURce:POWer:SPAN <num>](#).

Parameters

- <num> Any existing channel number. If unspecified, value is set to 1
- <num> Center power. Actual achievable leveled power depends on frequency.
- <port> If provided, this argument is **ignored** by the PNA.

Examples

```
SOUR:POW:CENT -15  
source2:power:center -7
```

Query Syntax SOURce<num>:POWer:CENTer?

Return Type Numeric

Default 0 dBm

SOURce<num>:POWer<port>:COUPle <ON | OFF>

(Read-Write) Turns Port Power Coupling ON or OFF.

Parameters

- <num> Any existing channel number. If unspecified, value is set to 1
- <ON | OFF> **ON** (or 1) - turns coupling ON. The same power level is used for both source ports.
OFF (or 0) - turns coupling OFF. Power level can be set individually for each source port.
- <port> If provided, this argument is **ignored** by the PNA.

Examples

```
SOUR:POW:COUP ON  
source2:power:couple off
```

Query Syntax SOURce<num>:POWer:COUPle?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON

SOURce<num>:POWer:DETector <char> **OBSOLETE**

(Read-Write) The PNA models with external leveling are now OBSOLETE.

Sets the source leveling loop as Internal or External.

Parameters

<num> Any existing channel number. If unspecified, value is set to 1

<char> **INT**ernal - Internal leveling is applied to the source

EXTernal - External leveling is applied to the source through a rear-panel connector. ONLY provided on 3 GHz, 6 GHz, and 9 GHz PNA models.

Examples

```
SOUR:POW:DET INT  
source2:power:detector external
```

Query Syntax SOURce<num>:POWER:DETector?

Return Type Character

Default INTernal

SOURce<num>:POWER<port>[:LEVel][:IMMediate][:AMPLitude] <num>, [src]

(Read-Write) Sets the RF power output level.

Parameters

<num> Any existing channel number. If unspecified, value is set to 1

<port> Source port number of the PNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.

<num> Source power in dBm.

Note: The range of settable power values depends on the PNA model and if source attenuators are installed. To determine the range of values, perform a query using MAX, then MIN, as an argument. ([SOUR:POW:ATT:AUTO](#) must be set to ON) Example: SOURce:POWER? Max

Actual achievable leveled power depends on frequency.

[src] **String**. (NOT case sensitive). Source port. Optional. Use [SOUR:CAT?](#) to return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an [external source](#), [true mode balanced port](#), or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

```
SOUR:POW1 5
source2:power:level:immediate:amplitude maximum
sour:pow 5, "Port 1 Src2"
```

Query Syntax SOURce<cnum>:POWER[:LEVel][:IMMediate][:AMPLitude]? [src]

Return Type Numeric

Default 0 dBm

SOURce<cnum>:POWER<port>[:LEVel]:SLOPe <num>

(Read-Write) Sets the RF power slope value.

Also enable the slope state using [SOUR:POW:SLOP:STAT ON](#).

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <num> Slope value in db/GHz. Choose any value between **-2** and **2** (0 is no slope).
- <port> If provided, this argument is **ignored** by the PNA.

Examples

```
SOUR:POW:SLOP .5234434
source2:power:level slope -1.345
```

Query Syntax SOURce<cnum>:POWER[:LEVel]:SLOPe?

Return Type Numeric

Default 0

SOURce<cnum>:POWER<port>[:LEVel]:SLOPe:STATe <ON | OFF>

(Read-Write) Turns Power Slope ON or OFF. Set the slope using [SOUR:POW:SLOP](#).

Parameters

- <num> Any existing channel number. If unspecified, value is set to 1
- <ON|OFF> **ON** (or 1) - turns slope ON.
- OFF** (or 0) - turns slope OFF.
- <port> If provided, this argument is **ignored** by the PNA.

Examples

```
SOUR:POW:SLOP:STAT ON  
source2:power:slope:state off
```

Query Syntax SOURce<num>:POWER[:LEVEL]:SLOPe:STATe?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF

SOURce<num>:POWER<port>:MODE <state>, [src]

(Read-Write) Sets the state of PNA source for the specified port.

Parameters

- <num> Any existing channel number. If unspecified, value is set to 1
- <port> Source port number of the PNA. If unspecified, <port> is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- <state> Source state. Choose from:
- **AUTO** Source power is turned ON when required for a measurement.
 - **ON** Source power is always ON regardless of the measurement.
 - **OFF** Source power is always OFF regardless of the measurement.

[src] **String**. (NOT case sensitive). Source port. Optional. Use [SOUR:CAT?](#) to return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an [external source](#), [true mode balanced port](#), or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

```
SOUR:POW:MODE ON
source2:power4:mode OFF
sour:pow:mode on, "Port 1 Src2"
```

Query Syntax SOURce<cnum>:POWer<port>:MODE? [src]**Return Type** Character**Default** Auto**SOURce<cnum>:POWer<port>:PORT:STARt <num>, [src]**

(Read-Write) Sets and reads the power sweep start power value for a specific port. This allows uncoupled forward and reverse power sweep ranges. Must also set [SENS:SWE:TYPE POWER](#) and [SOUR:POW:COUPLe OFF](#).

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <port> Source port number of the PNA. If unspecified, <port> is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- <num> Start power in dBm.

Note: The range of settable power values depends on the PNA model and if source attenuators are installed. To determine the range of values, perform a query using MAX, then MIN, as an argument. ([SOUR:POW:ATT:AUTO](#) must be set to ON) Example: SOURce:POWer:STARt? MIN

Actual achievable leveled power depends on frequency.

[src] **String.** (NOT case sensitive). Source port. Optional. Use [SOUR:CAT?](#) to return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an [external source](#), [true mode balanced port](#), or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

```
SOUR:POW1:PORT:STAR -15
source2:power:port:start 5, "bal port 1"
```

Query Syntax SOURce<cnum>:POWer<port>:PORT:STARt? [src]**Return Type** Numeric

Default -10 dBm

SOURce<cnum>:POWER<port>:PORT:STOP <num>, [src]

(Read-Write) Sets and reads the power sweep stop power value for a specific port. This allows uncoupled forward and reverse power sweep ranges. Must also set [SENS:SWE:TYPE POWER](#) and [SOUR:POW:COUPLE OFF](#).

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <port> Source port number of the PNA. If unspecified, <port> is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- <num> Stop power in dBm.

Note: The range of settable power values depends on the PNA model and if source attenuators are installed. To determine the range of values, perform a query using MAX, then MIN, as an argument. ([SOUR:POW:ATT:AUTO](#) must be set to ON) Example: SOURce:POWER:STARt? MIN

Actual achievable leveled power depends on frequency.

[src] **String.** (NOT case sensitive). Source port. Optional. Use [SOUR:CAT?](#) to return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an [external source](#), [true mode balanced port](#), or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

```
SOUR:POW1:PORT:STOP -15
source2:power:port:stop 5, "bal port 1"
```

Query Syntax SOURce<cnum>:POWER<port>:PORT:STOP? [src]

Return Type Numeric

Default 0 dBm

SOURce<cnum>:POWER<port>:SPAN <num>

(Read-Write) Sets the power sweep span power. Must also set:

[SENS:SWE:TYPE POWER](#) and [SOURCE:POWER:CENTER <num>](#).

Parameters

- <num> Any existing channel number. If unspecified, value is set to 1.
- <num> Span power. Actual achievable leveled power depends on frequency.
- <port> If provided, this argument is **ignored** by the PNA.

Examples

```
SOUR:POW:SPAN -15  
source2:power:span -7
```

Query Syntax SOURCE<num>:POWER:SPAN?

Return Type Numeric

Default 0 dBm

SOURCE<num>:POWER<port>:START <num>

(Read-Write) Sets the power sweep start power for ALL ports being used by the specified channel. Must also set:

[SENS:SWE:TYPE POWER](#) and [SOURCE:POWER:STOP <num>](#).

To set start power for a specific port, use [SOURCE:POW:PORT:START](#).

Parameters

- <num> Any existing channel number. If unspecified, value is set to 1
- <num> Start power.

Note: The range of settable power values depends on the PNA model and if source attenuators are installed. To determine the range of values, perform a query using MAX, then MIN, as an argument. ([SOURCE:POW:ATT:AUTO](#) must be set to ON) Example: SOURCE:POWER:START? MIN

Actual achievable leveled power depends on frequency.

- <port> If provided, this argument is **ignored** by the PNA.

Examples

```
SOUR:POW:STAR -15  
source2:power:start -7
```

Query Syntax SOURCE<num>:POWER:START?

Return Type Numeric

Default 0 dBm

SOURce<cnm>:POWER<port>:STOP <num>

(Read-Write) Sets the power sweep stop power for ALL ports being used by the specified channel.. Must also set: [SENS:SWE:TYPE POWER](#) and [SOURce:POWER:START <num>](#).

To set start power for a specific port, use [SOUR:POW:PORT:STOP](#).

Parameters

<cnm> Any existing channel number. If unspecified, value is set to 1

<num> Stop power.

Note: The range of settable power values depends on the PNA model and if source attenuators are installed. To determine the range of values, perform a query using MAX, then MIN, as an argument. ([SOUR:POW:ATT:AUTO](#) must be set to ON) Example: SOURce:POWER:STOP? MAX

Actual achievable leveled power depends on frequency.

<port> If provided, this argument is **ignored** by the PNA.

Examples

```
SOUR:POW:STOP -15
```

```
source2:power:stop -7
```

Query Syntax SOURce<cnm>:POWER:STOP?

Return Type Numeric

Default 0 dBm

Last modified:

11-Nov-2013	Mark detector in table red
30-Apr-2013	Edits to Sour:Pow:slop not integer
2-Mar-2012	Moved rec level commands
22-Dec-2011	Add ignored <port> argument
8-Jul-2011	Edit start and stop sweep commands
11-Jan-2011	Many minor edits
3-Dec-2010	Added link to source phase
16-Sep-2010	Added LSPC
17-Jun-2010	Fixed error in receiver mode - reference
19-Oct-2009	Fixed many broken links
15-Apr-2009	Fixed sour:pow:mode example
12-Feb-2009	Added receiver leveling
31-Jul-2008	Fix sour:pow unit in example
23-May-2008	Added uncoupled power sweep commands
25-Oct-2007	Edit test for source and rec attenuators commands
27-Jun-2007	Edited wording on Source:Cat?
10/18/06	MQQ Added Mode command

Source:Power:Correction Commands

Used to perform source power calibration on internal and external sources.

Note: Only ONE Source Power Cal can be performed at a time.

SOURce:POWer:CORRection

COLLect

- | **ABORt**
- | **ACQuire**
- | **AVERage**
 - | **[COUNT]**
 - | **NTOLerance**
- | **DISPlay**
 - | **[STATe]**
- | **FCHeck**
 - | **[STATe]**
- | **ITERation**
 - | **[COUNT]**
 - | **NTOLerance**
- | **METHod**
- | **SAVE**
- | **SENSor**
 - | **[FRANge]**
 - | **RCFactor**
 - | **SElect**
- | **TABLE**
 - | **DATA**
 - | **FREQuency**
 - | **LOSS**
 - | **[STATe]**
 - | **POINts?**
 - | **SElect**

DATA

- | **PRIOr**
- LEVel [AMPLitude]**
- OFFSet**
 - | **[MAGNitude]**
- [STATe]**

Click on a [blue](#) keyword to view the command details.

Red commands are superseded.

See Also

- Example program using these commands.
- Template for creating your own Power Meter Driver
- [Learn about Source Power Cal](#)
- Synchronizing the PNA and Controller
- SCPI Command Tree

Note : The `SOURCE:POWER:CORREction:COLLect:ACQuire` command, used to step the PNA and read a power meter, cannot be sent over the GPIB unless the power meter is connected to a different GPIB interface. See the alternative methods described in the command details.

`SOURCE<ch>:POWER<port>:CORREction:COLLect:ABORT`

(Write-only) Aborts a source power calibration sweep that is in progress.

To use this `ABORT` command, you **MUST** use the **ASYNchronous** argument with `SOUR:POW:CORR:COLL:ACQ`

After aborting, this message appears in the error log: **+243,"Requested operation was canceled"**.

Parameters

- `<ch>` Channel number of the source power cal. If unspecified, value is set to 1
- `<port>` Port number to correct for source power. If unspecified, value is set to 1.

Examples

```
SOUR:POW:CORR:COLL:ABOR
source1:power2:correction:collect:abort
```

Query Syntax Not Applicable

Default Not Applicable

`SOURCE<ch>:POWER<port>:CORREction:COLLect[:ACQuire] <char>,<id>[,src][,sync]`

Note: With PNA Rev. 6.2, a new "id" argument has been added to this command, replacing `SOUR:POW:CORR:COLL:METH` .

(Write-only) Initiates a source power cal acquisition sweep using the power sensor attached to the specified channel (A or B) on the power meter, using a USB power sensor, or using the specified PNA receiver.

For source power cal, the power meter can **NOT** be controlled by the PNA using the GPIB

Talker/Listener interface. Instead use one of the following methods:

- If present, use the GPIB dedicated controller port .
- Connect the power meter to the PNA using a USB / GPIB interface (Agilent 82357A).
- SCPI programming of the PNA using a LAN Client interface (see example).
- Send SCPI commands through the COM interface using the SCPI String Parser object.
- Directly control the Power Meter and PNA to step frequency; then acquire and store the Power reading. (see example).
- Configure the Power Meter/Sensor as a PMAR Device. Learn how . See SCPI commands .

Parameters

<ch> Channel number of the source power cal. If unspecified, value is set to 1

<port> Port number to correct for source power. If unspecified, value is set to 1.

<char> Acquisition Choose from:

- **PMETer** - Power Meter is used for all readings.
- **PMReceiver** - Power meter for the first iteration; then use the reference receiver for remaining readings if necessary (same as "fast iteration" box checked on dialog box)
- **RECeiver** - Use PNA measurement receiver for all readings.

<id> **String** (Not case sensitive). The power sensor or PNA receiver to use for measuring power.

For **PMETer** or **PMRECeiver** , choose from:

- **"ASENSOR"** or **"BSENSOR"** . For U series USB sensors, always specify **"ASENSOR"**

For **RECeiver** , choose from:

- Any PNA receiver to acquire readings using physical or logical receiver notation .
- Any configured PMAR device name. Learn more about PMAR Devices . See PMAR commands .

[src] Optional argument. **String** . (NOT case sensitive). Source port. Use SOUR:CAT? to return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed

to access ports that are not simple numbers, such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

[sync] If this argument is specified, must also specify [src].

Choose from:

- **SYNChronous** - Blocks SCPI commands during standard measurement (default behavior).
- **ASYNchronous** - Does NOT block SCPI commands during standard measurement.

Learn more about this argument

Examples

```
SOUR:POW:CORR:COLL PMET,"ASENSOR","Port 1",ASYN 'acquires power meter readings using the A sensor, source port 1, asynchronous.
sourc1:power2:correction:collect:acquire receiver,"a1"
'acquires source cal readings using the reference receiver for port 1.
```

Query Syntax Not Applicable

Default Not Applicable

SOURce<ch>:POWer<port>:CORRection:COLLect:AVERage[:COUNT] <num>

(Read-Write) This command, along with SOUR:POW:CORR:COLL:AVER:NTOLerance, allows for settling of the power sensor READINGS.

Specifies the maximum number of power readings that are taken at each stimulus point to allow for measurement settling. Each reading is averaged with the previous readings at that stimulus point.

When this average meets the Average:NTOLerance value or this number of readings has been made, the average is returned as the valid reading.

This setting is not necessary when using a PNA receiver (SOUR:POW:CORR:COLL REC) to make the measurement.

Learn more.

Parameters

- <ch> If provided, this argument is **ignored** by the PNA.
- <port> If provided, this argument is **ignored** by the PNA.

<num> Maximum number of readings to make to allow for settling. Choose any number between 3 and 1000.

Examples

```
SOUR:POW:CORR:COLL:AVER 2  
source:power:correction:collect:average:count 3
```

Query Syntax SOURce:POWer:CORRection:COLLect:AVERAge[:COUNt]?

Return Type Numeric

Default 3

SOURce<ch>:POWer<port>:CORRection:COLLect:AVERAge:NTOLerance <num>

(Read-Write) This command, along with SOUR:POW:CORR:COLL:AVER:COUNT , allows for settling of the power sensor READINGS.

Each power reading is averaged with the previous readings at each stimulus point. When the average meets this nominal tolerance value or the max number of readings has been made, the average is returned as the valid reading.

This setting is not necessary when using a PNA receiver (SOUR:POW:CORR:COLL REC) to make the measurement.

Learn more.

Parameters

- <ch> If provided, this argument is **ignored** by the PNA.
- <port> If provided, this argument is **ignored** by the PNA.
- <num> Power measurement settling tolerance value in dB. Choose any number between 0 and 5.

Examples

```
SOUR:POW:CORR:COLL:AVER:NTOL .05  
source1:power2:correction:collect:average:ntolerance .003
```

Query Syntax SOURce:POWer:CORRection:COLLect:AVERAge:NTOLerance?

Return Type Numeric

Default .050 dBm

SOURce<ch>:POWer<port>:CORRection:COLLect:DISPlay[:STATe] <ON | OFF>

(Read-Write) Enables and disables the display of power readings on the PNA screen. Send this command BEFORE you begin a source power cal acquisition. After the source power cal data is acquired, this setting is reset to ON.

Parameters

- <ch> Channel number of the source power cal. If unspecified, value is set to 1
- <port> If provided, this argument is **ignored** by the PNA.
- <ON|OFF> **ON (1)** Source power calibration dialog box is shown on the PNA screen. Power readings are plotted against the Tolerance value as limit lines.
- OFF (0)** - Source power calibration dialog box is NOT shown on the PNA screen.

Examples

```
SOUR:POW:CORR:COLL:DISP ON  
source1:power2:correction:collect:display:state off
```

Query Syntax SOURce:POWer:CORRection:COLLect:DISPlay[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default ON (1)

SOURce<ch>:POWer<port>:CORRection:COLLect:FCHeck[:STATe] <ON | OFF>

(Read-Write) Enables and disables frequency checking of source power cal acquisition sweeps. ONLY use when you have more than one power sensor.

Parameters

- <ch> Channel number of the source power cal. If unspecified, value is set to 1
- <port> If provided, this argument is **ignored** by the PNA.
- <ON|OFF> **ON (1)** turns source power cal frequency checking ON. A requested acquisition will only succeed for those frequency points which fall within a frequency range specified for the power sensor being used. An acquisition will pause in mid-sweep if the frequency is about to exceed the maximum frequency limit specified for that sensor. When the sweep is paused in this manner, a sensor connected to the other channel input of the power meter can be connected to the measurement port in place of the previous sensor, and used to complete the sweep. However, the maximum frequency specified for the second sensor would need to be sufficient for the sweep to complete. Frequency limits are specified using the SOUR:POW:CORR:COLL:SEN command.
- OFF (0)** - turns source power cal frequency checking OFF. An acquisition will

use just one power sensor for the entire sweep, regardless of frequency.

Examples

```
SOUR:POW:CORR:COLL:FCH ON  
source1:power2:correction:collect:fcheck:state off
```

Query Syntax SOURce:POWer:CORRection:COLLect:FCHeck[:STATe]?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF (0)

SOURce<ch>:POWer<port>:CORRection:COLLect:ITERation[:COUNT] <num>

(Read-Write) This command, along with SOUR:POW:CORR:COLL:ITER:NTOL describes the number of adjustments to make to the source power.

Sets the maximum number of readings to take at each data point for iterating the source power. Power READINGS (performed by SOUR:POW:CORR:COLL:AVER:COUNT and SOUR:POW:CORR:COLL:AVER:NTOLerance) will continue to be made, and source power adjusted, until the measurement is within the iteration tolerance value or this max number of measurements has been met. The last value is the valid measurement for that data point.

Learn more.

Parameters

- <ch> If provided, this argument is **ignored** by the PNA.
- <port> If provided, this argument is **ignored** by the PNA.
- <num> Maximum number of readings. Choose any number between 1 and 1000.

Examples

```
SOUR:POW:CORR:COLL:ITER 2  
source:power:correction:collect:iteration 3
```

Query Syntax SOURce:POWer:CORRection:COLLect:ITERation[:COUNT]?

Return Type Numeric

Default 1

SOURce<ch>:POWer<port>:CORRection:COLLect:ITERation:NTOLerance <num>

(Read-Write) This command, along with SOUR:POW:CORR:COLL:ITER:COUNT describes the number of adjustments to make to the source power.

Sets the maximum desired deviation from the sum of the test port power and the offset value. Power READINGS (performed by SOUR:POW:CORR:COLL:AVER:COUNT and SOUR:POW:CORR:COLL:AVER:NTOLerance) will continue to be made, and source power adjusted, until a measurement is within this tolerance value or the max number of measurements has been met. The last value is the valid measurement for that data point.

Learn more.

Parameters

- <ch> If provided, this argument is **ignored** by the PNA.
- <port> If provided, this argument is **ignored** by the PNA.
- <num> Tolerance value in dBm. Choose any number between 0 and 5

Examples

```
SOUR:POW:CORR:COLL:ITER:NTOL .005  
source:power:correction:collect:iteration:ntolerance .1
```

Query Syntax SOURce:POWer:CORRection:COLLect:ITERation:NTOLerance?

Return Type Numeric

Default .05

SOURce<ch>:POWer<port>:CORRection:COLLect:METhod <char> **Superseded**

This command is replaced with SOUR:POW:CORR:COLLect[:ACQuire] which now specifies the method **and** the device. The only parameter required by that command was either **ASENSor** or **BSENSor** which are still supported but not documented.

(Read-Write) Selects the calibration method to be used for the source power cal acquisition.

Parameters

- <ch> Channel number of the source power cal. If unspecified, value is set to 1
- <port> Port number to correct for source power. If unspecified, value is set to 1.
- <char> Choose from:

NONE - No Cal method

PMETer - Power Meter is used for all readings. (same as "fast iteration" box not checked on dialog box)

PMReceiver - Power meter for the first iteration; then use the reference

receiver for remaining readings if necessary (same as "fast iteration" box checked on dialog box)

Examples

```
SOUR:POW:CORR:COLL:METH PMET  
source1:power2:correction:collect:method pmreceiver
```

Query Syntax SOURce:POWer:CORRection:COLLect:METhod?

Return Type Character

Default NONE

SOURce<ch>:POWER<port>:CORRection:COLLect:SAVE [<RREC>]

(Write-only) Applies the array of correction values after a source power calibration sweep has completed. The source power correction will then be active on the specified source port for channel <ch>. This command does NOT save the correction values. To save correction values, save an instrument / calibration state (*.cst file) after performing a source power cal.

Parameters

- <ch> If provided, this argument is **ignored** by the PNA.
- <port> If provided, this argument is **ignored** by the PNA.
- <RREC> Optional argument.

RRECeiver In addition to a source Power Cal, perform a calibration of the reference receiver used in the measurement. ONLY the Reference Receiver calibration is then saved to a Cal Set or Cal Register as specified by the current setting of SENS:CORR:PREF:CSET:SAVE .

This argument only applies to standard S-parameter channels.

Examples

```
SOUR:POW:CORR:COLL:SAVE  
source:power:correction:collect:save rreceiver
```

Query Syntax Not Applicable

Default Not Applicable

**SOURce<ch>:POWER<port>:CORRection:COLLect:<pmChan>SENsor[:FRANge]
<num1>,<num2>**

(Read-Write) Specifies the frequency range over which the power sensors connected to the specified channels (A and B) of the power meter can be used (minimum frequency, maximum frequency). If the power meter has only a single channel, that channel is considered channel A.

Parameters

- <ch> Channel number of the source power cal. If unspecified, value is set to 1
- <port> If provided, this argument is **ignored** by the PNA.
- <pmChan> Power Meter channel. Choose from:
- A** - Channel A
 - B** - Channel B
- <num1> Minimum frequency for the sensor. If a frequency unit is not specified, Hz is assumed. No limits are placed on this value.
- <num2> Maximum frequency for the sensor. If a frequency unit is not specified, Hz is assumed. No limits are placed on this value.

Examples

```
SOUR:POW:CORR:COLL:ASEN 100E3, 3E9  
source1:power2:correction:collect:bsensor:frange 10 MHz, 18 GHz
```

Query Syntax SOURce:POWer:CORRection:COLLect:ASENsor[:FRANge]?
SOURce:POWer:CORRection:COLLect:BSEnSor[:FRANge]?

Return Type Numeric

Default 0,0

SOURce<ch>:POWer<port>:CORRection:COLLect:<pmChan>SENsor:RCFactor <num>

(Read-Write)) Specifies the reference cal factor for the power sensor connected to channel A or B of the power meter. If the power meter has only a single channel, that channel is considered channel A.

Note : If the sensor connected to the specified channel of the power meter contains cal factors in EPROM (such as the Agilent E-series power sensors), those will be the cal factors used during the calibration sweep. The reference cal factor value associated with this command, and any cal factors entered into the PNA for that sensor channel, will not be used.

Parameters

- <ch> Channel number of the source power cal. If unspecified, value is set to 1
- <port> If provided, this argument is **ignored** by the PNA.
- <pmChan> Power Meter channel. Choose from:
 - A** - Channel A
 - B** - Channel B
- <num> Reference cal factor in percent. Choose any number between 1 and 150.

Examples

```
SOUR:POW:CORR:COLL:ASEN:RCF 98.7  
source1:power2:correction:collect:bsensor:rcfactor 105
```

Query Syntax SOURce:POWer:CORRection:COLLect:ASENsor:RCFactor?

SOURce:POWer:CORRection:COLLect:BSEnSor:RCFactor?

Return Type Numeric

Default 100

SOURce<ch>:POWer<port>:CORRection:COLLect:<pmChan>SENsor:SElect

(Read-Write) Sets and returns the power sensor channel (A or B) to be used. This performs the same function as the **Use this sensor only** checkbox in the Power Sensor Settings dialog .

Notes:

- This command is NOT necessary when performing a Guided Power Cal using Multiple Sensors.
- This command can be used with Application channels .

Parameters

- <ch> Channel number of the source power cal. If unspecified, value is set to 1
- <port> If provided, this argument is **ignored** by the PNA.
- <pmChan> Power Meter channel. Choose from:
- A - Channel A
 - B - Channel B**

Examples

```
SOUR:POW:CORR:COLL:<pmChan>SEN:SEL 'Write  
source1:power2:correction:collect:bsensor:select? 1e9 'Read
```

Query Syntax SOURce:POWer:CORRection:COLLect:ASENsor:SELEct? <Frequency>

SOURce:POWer:CORRection:COLLect:BSEnSor:SELEct? <Frequency>

Returns a boolean 1 or 0 (ON or OFF) indicating whether the sensor is to be used at the specified frequency.

If frequency checking is OFF, then the <Frequency> parameter is ignored. The query returns if the sensor is selected for ALL frequencies.

Return Type Numeric

Default Not Applicable

SOURce<ch>:POWer<port>:CORRection:COLLect:TABLE:DATA <data>

(Read-Write) Read or write data into the selected table. Use SOUR:POW:CORR:COLL:TABL:SElect to select a table.

- When the power sensor table is selected, the data is interpreted as cal factors in **percent** .
- When the loss table is selected, POSITIVE values in dB are interpreted as LOSS. To compensate for gain, use negative values.
- Each table can contain up to 9999 segments. Values can be loaded using the Characterize Adapter macro .
- Learn more about Power Loss Compensation .

Parameters

- <ch> Channel number of the source power cal. If unspecified, value is set to 1
- <port> If provided, this argument is **ignored** by the PNA.
- <data> Data to write into the selected table.

Examples

```
SOURce:POWer:CORREction:COLLect:TABLE:DATA 0.12, 0.34, 0.56
```

Query Syntax SOURce<ch>:POWer:CORREction:COLLect:TABLE:DATA?

If the selected table is currently empty, no data is returned.

Return Type Numeric - one number per table segment.

Default Not Applicable

SOURce<ch>:POWer<port>:CORREction:COLLect:TABLE:FREQuency <data>

(Read-Write) Read or write frequency values for the selected table (cal factor table for a power sensor, or the loss compensation table). Use SOUR:POW:CORR:COLL:TABL:SElect to select a table.

Parameters

- <ch> Channel number of the source power cal. If unspecified, value is set to 1
- <port> If provided, this argument is **ignored** by the PNA.
- <data> Frequency data to write into the selected table.

Examples

```
SOURce:POWer:CORREction:COLLect:TABLE:FREQuency 10E6, 1.5E9, 9E9
```

Query Syntax SOURce<ch>:POWer:CORREction:COLLect:TABLE:FREQuency?

If the selected table is currently empty, no data is returned.

Return Type Numeric - one number per table segment

Default Not Applicable

SOURce<ch>:POWER<port>:CORRection:COLLect:TABLE:LOSS[:STATE] <ON | OFF>

(Read-Write) Indicates whether or not to adjust the power readings using the values in the loss table during a source power cal sweep. Learn more about Power Loss Compensation .

Parameters

- <ch> Channel number of the source power cal. If unspecified, value is set to 1
- <port> If provided, this argument is **ignored** by the PNA.
- <ON|OFF> **ON (or 1)** - turns use of the loss table ON.
- OFF (or 0)** - turns use of the loss table OFF.

Examples

```
SOUR:POW:CORR:COLL:TABL:LOSS ON
source1:power2:correction:collect:table:loss:state off
```

Query Syntax SOURce:POWER:CORRection:COLLect:TABLE:LOSS[:STATE]?

Return Type Boolean (1 = ON, 0 = OFF)

Default OFF (0)

SOURce<ch>:POWER<port>:CORRection:COLLect:TABLE:POINTs?

(Read-only) Returns the number of segments that are currently in the selected table.

Parameters

- <ch> Channel number of the source power cal. If unspecified, value is set to 1
- <port> If provided, this argument is **ignored** by the PNA.

Examples

```
SOUR:POW:CORR:COLL:TABL:POIN?
source1:power2:correction:collect:table:points?
```

Return Type Numeric

Default 0

SOURce<ch>:POWER<port>:CORRection:COLLect:TABLE[:SElect] <char>

(Read-Write) Selects which table you want to write to or read from. Read or write using SOURce:POWer:CORRection:COLLect:TABLE:FREQUency and SOURce:POWer:CORRection:COLLect:TABLE:DATA

Parameters

- <ch> Channel number of the source power cal. If unspecified, value is set to 1
- <port> If provided, this argument is **ignored** by the PNA.
- <char> Choose from:

NONE - No table selected

ASENSor - Cal Factor table for Power Sensor A

BSENSor - Cal Factor table for Power Sensor B

LOSS - Loss compensation table

Examples

```
SOUR:POW:CORR:COLL:TABL ASEN  
source1:power2:correction:collect:table:select bsensor
```

Query Syntax SOURce:POWer:CORRection:COLLect:TABLE[:SElect]?

Return Type Character

Default NONE

SOURce<ch>:POWer<port>:CORRection:DATA <data>[,src]

(Read-Write) Writes and reads source power calibration data.

The effect from this command on the channel is immediate. Do NOT send SOUR:POW:CORR:COLL:SAVE after this command as it may invalidate the uploaded data.

When querying source power calibration data, if no source power cal data exists for the specified channel and source port, then no data is returned.

If a change in the instrument state causes interpolation and/or extrapolation of the source power cal, the correction data associated with this command correspond to the new instrument state (interpolated and/or extrapolated data).

If the channel is sweeping the source backwards, then the first data point is the highest frequency value; the last data point is the lowest. Use the SENS:X:VALues? command to return the X-axis values in the displayed order.

Parameters

- <ch> Channel number of the source power cal. If unspecified, value is set to 1
- <port> Port number to correct for source power. If unspecified, value is set to 1.
- <data> Correction Data

[src] **String** . (NOT case sensitive). Source port. Optional. Use SOUR:CAT? to return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed to access ports that are not simple numbers, such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

```
SOURce1:POWer2:CORRection:DATA 0.12, -0.34, 0.56
```

Query Syntax SOURce<ch>:POWer<port>:CORRection:DATA? [src]

Return Type Depends on FORMat:DATA command

Default Not Applicable

SOURce<ch>:POWer<port>:CORRection:DATA:PRIor <data>[,src]

(Read-Write) Writes and reads power correction values from the previous iteration of the source power cal. Data for which the first power meter reading were within the tolerance limit, the prior correction value is 0.

In all other respects, this command is the same as SOUR:POW:CORR:DATA .

This command can be used to determine the final power reading at each point of the power cal, for a cal that did not pass tolerance limits. The formula for determining the power reading (in dB):

Power reading = Target power at the source port + specified power cal offset value + prior iteration corr value actual power corr value.

The "actual" value in this equation is returned with SOUR:POW:CORR:DATA?

Parameters

- <ch> Channel number of the source power cal. If unspecified, value is set to 1
- <port> Port number to correct for source power. If unspecified, value is set to 1.
- <data> Correction Data

[src] **String** . (NOT case sensitive). Source port. Optional. Use SOUR:CAT? to return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed

to access ports that are not simple numbers, such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

```
SOURce1:POWer2:CORRection:DATA:PRIor 0.12, -0.34, 0.56
```

Query Syntax SOURce<ch>:POWer<port>:CORRection:DATA:PRIor? [src]

Return Type Depends on FORMat:DATA command

Default Not Applicable

SOURce<ch>:POWer<port>:CORRection:LEVel[:AMPLitude] <num>[,src]

(Read-Write) Specifies the power level that is expected at the desired reference plane (DUT input or output). This is not used for segment sweep with independent power levels or power sweeps .

Note: Although this command still works, it is recommended that you specify cal power by setting the test port power and offset value.

Parameters

- <ch> Channel number of the source power cal. If unspecified, value is set to 1
- <port> Port number to correct for source power. If unspecified, value is set to 1.
- <num> Cal power level in dBm. Because this could potentially be at the output of a device-under-test, no limits are placed on this value here. It is realistically limited by the specifications of the device (power sensor) that will be used for measuring the power. The power delivered to the PNA receiver must never exceed PNA specifications for the receiver!
- [src] **String** . (NOT case sensitive). Source port. Optional. Use SOUR:CAT? to return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed to access ports that are not simple numbers, such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

```
SOUR:POW:CORR:LEV 10  
source1:power2:correction:level:amplitude 0 dbm
```

Query Syntax SOURce:POWer:CORRection:LEVel[:AMPLitude]? [src]

Return Type Numeric

Default 0 dBm

SOURce<ch>:POWER<port>:CORRection:OFFSet[:MAGNitude] <num>[,src]

(Read-Write) Sets or returns a power level offset from the PNA test port power. This can be a gain or loss value (in dB) to account for components you connect between the source and the reference plane of your measurement. For example, specify 10 dB to account for a 10 dB amplifier at the input of your DUT.

Cal power is the sum of the test port power setting and this offset value. Following the calibration, the PNA power readouts are adjusted to the cal power.

Parameters

- <ch> Channel number of the source power cal. If unspecified, value is set to 1
- <port> Port number to correct for source power. If unspecified, value is set to 1.
- <num> Gain or loss value in dB. Choose a value between -200 and 200
- [src] **String** . (NOT case sensitive). Source port. Optional. Use SOUR:CAT? to return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed to access ports that are not simple numbers, such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

```
SOUR:POW:CORR:OFFS 10  
source1:power2:correction:offset:magnitude -3
```

Query Syntax SOURce:POWER:CORRection:OFFSet[:MAGNitude]? [src]

Return Type Numeric

Default 0 dB

SOURce<ch>:POWER<port>:CORRection[:STATe] <bool>[,src]

(Read-Write) Enables and disables source power correction for the specified port on the specified channel.

Parameters

- <ch> Channel number of the source power cal. If unspecified, value is set to 1
- <port> Port number to correct for source power. If unspecified, value is set to 1.
- <bool> ON (or 1) turns source power correction ON.
OFF (or 0) - turns source power correction OFF.
- [src] **String** . (NOT case sensitive). Source port. Optional. Use SOUR:CAT? to return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed to access ports that are not simple numbers, such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

```
SOUR:POW:CORR ON  
sourc1:power2:correction:state off
```

Query Syntax SOURce:POWer:CORRection[:STATe]? [src]

Return Type Boolean (1 = ON, 0 = OFF)

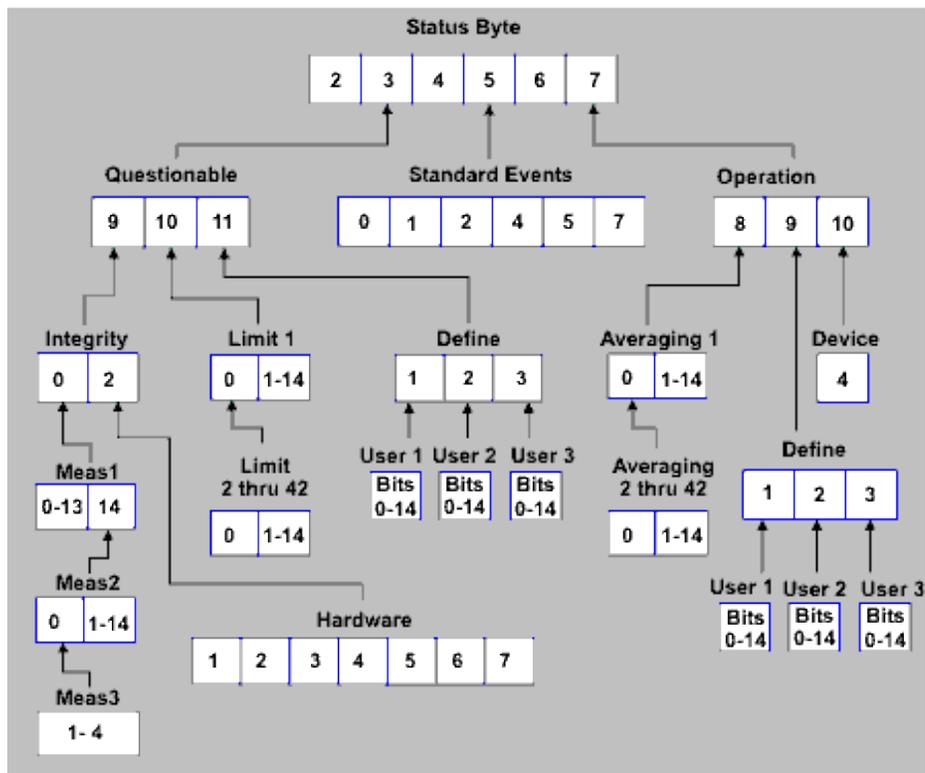
Default OFF (0)

Last modified:

25-Nov-2013	Updated ABORt command (BH)
11-Nov-2013	Corrected a couple of heading arrangements
8-Nov-2012	Increase readings and settling max
23-Aug-2012	Fixed note for Sensor:Sel (BH)
9-Aug-2012	Fixed typo. Thanks LS
14-May-2012	Increase limit in Loss table
11-Apr-2012	Edited ACQ connection methods
22-Dec-2011	Added ignored arguments
23-Aug-2011	Fixed typo. Thx Brad
10-Feb-2011	Added PMAR to ACQUIRE command
25-Oct-2010	Added immediate note to DATA command
17-Mar-2010	Added PRIor command (9.2)
24-Feb-2009	Added sync to ACQ, removed old arguments
2-Jun-2008	Clarified Loss compensation data
17-Apr-2007	Removed ch and port arguments for 4 settling and accuracy commands.
12-Sept-2006	MQ Modified for receiver only SPC

Status Register Commands

The status registers enable you to query the state of selected events that occur in the analyzer.



Note: This documentation requires familiarity with the "Standard Status Data Structure - Register Model" as defined in IEEE Std 488.2-1992. Also, first read [Learn about Status Registers](#)

Click on a [blue](#) keyword to view the command details.

See Also

- [Example Programs](#)
- [Learn about Status Registers](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

Notes:

- Any bit not shown in the registers is not used but may be reserved for future use.
- The [SCPIStringParser](#) can NOT be used with SCPI Status Reporting. However, the *OPC? will work.

Status Byte Register

Summarizes the states of the other registers and monitors the PNA output queue. It also generates **service requests**. The Enable register is called the Service Request Enable Register.

Commands Description

- *CLS Clears ALL "event" registers and the SCPI Error / Event queue. The corresponding ENABLE registers are unaffected.
- *STB? Reads the value of the analyzer's status byte. The byte remains after being read.
- *SRE? Reads the current state of the Service Request **Enable** Register.
- *SRE <num> Sets bits in the Service Request **Enable** register. The current setting of the SRE register is stored in non-volatile memory. Use *SRE 0 to clear the enable.

<num> Combined value of the weights for bits to be set.

Bit	Weight	Description	Bit is set to 1 when the following conditions exist:
2	4	Error / Event queue Summary (EAV)	the Error / Event queue is not empty. To read the error message, use SYST:ERR?
3	8	Questionable Register Summary	any enabled bit in the questionable event status register is set to 1
4	16	Message Available	the output queue is not empty
5	32	Standard Event Register Summary	any enabled bit in the standard event status register is set to 1
6	64	Request Service	any of the other bits in the status byte register is set to 1 (used to alert the controller of a service request within the analyzer). This bit cannot be disabled.
7	128	Operation Register Summary	any enabled bit in the standard operation event status register is set to 1

STATus:QUEStionable:<keyword>

Summarizes conditions that monitor the quality of measurement data.

[<keyword>](#) Example

:CONDition? STAT:QUES:COND?

:ENABle <bits> STAT:QUES:ENAB 1024

<bits>

[:EVENTt]? STAT:QUES?

:NTRansition <bits> STAT:QUES:NTR 1024

<bits>

:PTRansition <bits> STAT:QUES:PTR 0

<bits>

Bit	Weight	Description	Bit is set to 1 when the following conditions exist:
9	512	Integrity Reg summary	any enabled bit in the Integrity event register is set to 1
10	1024	Limit Registers summary	any enabled bit in the Limit event registers is set to 1
11	2048	Define Registers summary	any enabled bit in the Define event registers is set to 1

STATus:QUEStionable:INTegrity <keyword>

Summarizes conditions in the Measurement Integrity register.

[<keyword>](#) Example

:CONDition? STAT:QUES:INT:COND?

:ENABle <bits> STAT:QUES:INT:ENAB 1024

[:EVENTt]? STAT:QUES:INT?

:NTRansition <bits> STAT:QUES:INT:NTR 1024

:PTRansition <bits> STAT:QUES:INT:PTR 0

Bit	Weight	Description	Bit is set to 1 when the following conditions exist:
0	1	Measurement Summary	any bit in the Measurement Integrity event register is set to 1
2	4	Hardware Summary	any bit in the Hardware event register is set to 1

STATus:QUEStionable:INTegrity:HARDware<keyword>

Monitors the status of hardware failures.

<keyword> Example

:CONDition? STAT:QUES:INT:HARD:COND?

:ENABle <bits> STAT:QUES:INT:HARD:ENAB 1024

[:EVENT]? STAT:QUES:INT:HARD?

:NTRansition <bits> STAT:QUES:INT:HARD:NTR 1024

:PTRansition <bits> STAT:QUES:INT:HARD:PTR 0

Bit	Weight	Description	Bit is set to 1 when the following conditions exist:
1	2	Phase Unlock	the source has lost phaselock, possibly caused by a reference channel open or a hardware failure.
2	4	Unleveled	the source power is unleveled. This could be caused by a source set for more power than it can deliver at the tuned frequency. Or it could be caused by a hardware failure.
3	8	Not used	N/A
4	16	EE Write Failed	an attempted write to the EEPROM has failed, possibly caused by a hardware failure.
5	32	Not used	N/A
6	64	Ramp Cal Failed	the analyzer was unable to calibrate the analog ramp generator due to a possible hardware failure.
7	128	Not used	N/A

STATus:QUEStionable:INTEgrity:MEASurement<n> <keyword>

Note: This register can be used ONLY with standard S-parameter measurements.

Monitors the lag between changing a channel setting and when the data is ready to query.

When you change the channel state (start/stop freq, bandwidth, and so forth), then the questionable bit for that channel is set. This indicates that your desired channel state does not yet match the data you would get if querying a data trace. When the next sweep is complete (without aborting in the middle), and the data trace matches the channel state that produced it, the bit is cleared for that channel.

<n> Measurement register number. Choose from 1 to 3

<keyword> **Example**

:CONDition? STAT:QUES:INT:MEAS1:COND?

:ENABle <bits> STAT:QUES:INT:MEAS2:ENAB 1024

[:EVENT]? STAT:QUES:INT:MEAS3?

:NTRansition <bits> STAT:QUES:INT:MEAS2:NTR 1024

:PTRansition <bits> STAT:QUES:INT:MEAS1:PTR 0

		Measurement Register <n>			
Bit	Weight	1	2	3	Bit is set to 1 when the following conditions exist:
0	1	1	Summary from Meas Reg 3		a setting change on this channel has occurred and the data does not yet reflect that change.
1	2	2	15	29	a setting change on this channel has occurred and the data does not yet reflect that change.
2	4	3	16	30	a setting change on this channel has occurred and the data does not yet reflect that change.
3	8	4	17	31	a setting change on this channel has occurred and the data does not yet reflect that change.
4	16	5	18	32	a setting change on this channel has occurred and the data does not yet reflect that change.
5	32	6	19		a setting change on this channel has occurred and the data does not yet reflect that change.
6	64	7	20		a setting change on this channel has occurred and the data does not yet reflect that change.

7	128	8	21		a setting change on this channel has occurred and the data does not yet reflect that change.
8	256	9	22		a setting change on this channel has occurred and the data does not yet reflect that change.
9	512	10	23		a setting change on this channel has occurred and the data does not yet reflect that change.
10	1024	11	24		a setting change on this channel has occurred and the data does not yet reflect that change.
11	2048	12	25		a setting change on this channel has occurred and the data does not yet reflect that change.
12	4096	13	26		a setting change on this channel has occurred and the data does not yet reflect that change.
13	8192	14	27		a setting change on this channel has occurred and the data does not yet reflect that change.
14	16384	Summary from Meas Reg 2	28		a setting change on this channel has occurred and the data does not yet reflect that change.

STATus:QUEStionable:LIMit<n> <keyword>

Monitors and summarizes the status of limit line failures. When a trace fails, the representative bit is set to 1.

Bit 0 is used to summarize failures in the registers that follow. For example, Limit Register 3, bit 0, summarizes the failures from registers 4 through 42.

All enable bits are set to 1 by default.

To find the measurement number, use [Calc:Par:Mnum](#)

<n> Limit register: Choose from 1 to 42.

<keyword> **Example**

:CONDition? **STAT:QUES:LIM4:COND?**

:ENABle <bits> **STAT:QUES:LIM1:ENAB 1024**

[:EVENTt]? **STAT:QUES:LIM3?**

:NTRansition <bits> **STAT:QUES:LIM2:NTR 1024**

:NTRansition? **STAT:QUES:LIM1:NTR?**

:PTRansition <bits> STAT:QUES:LIM5:PTR 0

:PTRansition? STAT:QUES:LIM1:PTR?

		Limit Register <n>											
Bit	Weight	1	2	3	4	5	6	7	8	...	41	42	Bit is set to 1 when the following conditions exist:
0	1	2-42	3-42	4-42	5-42	6-42	7-42	8-42	9-42	...	42	--	Summary Bit - If any bit from that register fails, it propagates to the previous register, bit 0.
		Trace Numbers											
1	2	1	15	29	43	57	71	85	99	...	561	575	any point on trace fails the limit test
2	4	2	16	30	44	58	72	86	100	...	562	576	any point on trace fails the limit test
3	8	3	17	31	45	59	73	87	101	...	563	577	any point on trace fails the limit test
4	16	4	18	32	46	60	74	88	102	...	564	578	any point on trace fails the limit test
5	32	5	19	33	47	61	75	89	103	...	565	579	any point on trace fails the limit test
6	64	6	20	34	48	62	76	90	104	...	566	580	any point on trace fails the limit test
7	128	7	21	35	49	63	77	91	105	...	567	--	any point on trace fails the limit test
8	256	8	22	36	50	64	78	92	106	...	568	--	any point on trace fails the limit test
9	512	9	23	37	51	65	79	93	107	...	569	--	any point on trace fails the limit test
10	1024	10	24	38	52	66	80	94	108	...	570	--	any point on trace fails the limit test
11	2048	11	25	39	53	67	81	95	109	...	571	--	any point on trace fails the limit test

12	4096	12	26	40	54	68	82	96	110	...	572	--	any point on trace fails the limit test
13	8192	13	27	41	55	69	83	97	111	...	573	--	any point on trace fails the limit test
14	16384	14	28	42	56	70	84	98	112	...	574	--	any point on trace fails the limit test

To determine Register, Bit number, and Weight for trace numbers between 113 and 560 (not shown in the above table) use the following calculations.

The limit status for trace numbers higher than 580 can NOT be tracked.

The following example calculates the Register, Bit number, and Bit Weight for trace # 400:

- To determine Limit **Register** number, use $((\text{Trace \#} - 1) / 14) + 1$.
- To determine Limit **Bit Number**, use the **remainder** +1 of the above calculation.
- $((400-1)/14) + 1 = \text{Register\# } r+1\text{Bit}$
 - $399/14 = 28 \text{ r}7$
 - $28+1= \text{Register } 29$
 - $7+1= \text{Bit number } 8$
- To determine Limit **Bit Weight**: Use above table. For example: Bit 8 = **256**

STATus:QUEStionable:DEFine<keyword>

Summarizes conditions in the Questionable:Define:User<1|2|3> event registers.

<keyword>	Example
:CONDition?	STAT:QUES:DEF:COND?
:ENABle <bits>	STAT:QUES:DEF:ENAB 1024
[:EVENT]?	STAT:QUES:DEF?
:NTRansition <bits>	STAT:QUES:DEF:NTR 1024
:PTRansition <bits>	STAT:QUES:DEF:PTR 0

Bit	Weight	Description	Bit is set to 1 when the following conditions exist:
1	2	USER1	any bit in the USER1 event register is set to 1
2	4	USER2	any bit in the USER2 event register is set to 1
3	8	USER3	any bit in the USER3 event register is set to 1

STATus:QUEStionable:DEFine:USER<1|2|3><keyword>

Monitors conditions that you define and map in any of the three QUES:DEF:USER event registers.

<keyword> Example

:ENABLE <bits> STAT:QUES:DEF:USER1:ENABle 1024

[:EVENT]? STAT:QUES:DEF:USER1?

:MAP <bit>,<error> STAT:QUES:DEF:USER1:MAP 0,-113 'when error -113 occurs, bit 0 in USER1 will set to 1.

Bit	Weight	Description	Bit is set to 1 when the following conditions exist:
0	1	for user	user defined
1	2	for user	user defined
2	4	for user	user defined
3	8	for user	user defined
4	16	for user	user defined
5	32	for user	user defined
6	64	for user	user defined
7	128	for user	user defined
8	256	for user	user defined
9	512	for user	user defined
10	1024	for user	user defined
11	2048	for user	user defined
12	4096	for user	user defined
13	8192	for user	user defined
14	16384	for user	user defined

Standard Event Status Register

Monitors "standard" events that occur in the analyzer. This register can only be cleared by:

- a Clear Command (*CLS).
- reading the Standard Enable Status Register (*ESE?).
- a power-on transition. The analyzer clears the register and then records any transitions that occur, including setting the Power On bit (7).

Commands Description

*ESE? Reads the settings of the standard event **ENABLE** register.

*ESE <*bits*> Sets bits in the standard event **ENABLE** register. The current setting is saved in non-volatile memory.

<*bits*> The sum of weighted bits in the register. Use *ESE 0 to clear the enable register.

*ESR? Reads and clears the **EVENT** settings in the Standard Event Status register.

*OPC Sets bit 0 when the overlapped command is complete. (see Understanding Command Synchronization / OPC).

*OPC? Operation complete query - read the Operation Complete bit (0).

Bit	Weight	Description	Bit is set to 1 when the following conditions exist:
0	1	Operation Complete	the two following events occur in order : 1. the *OPC command is sent to the analyzer 2. the analyzer completes all pending overlapped commands
1	NA	Request Control	Not Supported - the analyzer application is not configured to control GPIB operation
2	4	Query Error	a query error is detected indicating: - an attempt to read data from the output queue when no data was present OR - data in the output queue was lost, as in an overflow
4	16	Execution Error	an execution error is detected indicating: - a <PROGRAM DATA> element was outside the legal range or inconsistent with the operation of the analyzer OR

			- the analyzer could not execute a valid command due to some internal condition
5	32	Command Error	a command error is detected indicating that the analyzer received a command that: <ul style="list-style-type: none"> • did not follow proper syntax • was misspelled • was an optional command it does not implement
7	128	Power ON	Power to the analyzer has been turned OFF and then ON since the last time this register was read.

STATus:OPERation<keyword>

Summarizes conditions in the Averaging and Operation:Define:User<1|2|3> event registers.

<keyword> **Example**

:CONDition? **STAT:OPER:COND?**

:ENABle <bits> **STAT:OPER:ENAB 1024**

[:EVENT]? **STAT:OPER?**

:NTRansition <bits> **STAT:OPER:NTR 1024**

:PTRansition <bits> **STAT:OPER:PTR 0**

Bit	Weight	Description	Bit is set to 1 when the following conditions exist:
8	256	Averaging summary	either enabled bit in the Averaging summary event register is set to 1
9	512	User Defined summary	
10	1024	Device summary	either enabled bit in the Device summary event register is set to 1

STATus:OPERation:AVERaging<n> <keyword>

Monitors and summarizes the status of Averaging on traces 1 to 580. When averaging for a trace is complete, the representative bit is set to 1.

Bit 0 is used to summarize the status in the registers that follow. For example, Average Register 3, bit 0, summarizes the status from registers 4 through 42.

All enable bits are set to 1 by default.

To find the measurement number, use [Calc:Par:Mnum](#).

<n> Averaging Register. Choose from 1 to 42

<keyword> **Example**

:CONDition?	STAT:OPER:AVER1:COND?
:ENABle <bits>	STAT:OPER:AVER1:ENAB 1024
[:EVENT]?	STAT:OPER:AVER1?
:NTRansition <bits>	STAT:OPER:AVER1:NTR 1024
:PTRansition <bits>	STAT:OPER:AVER1:PTR 0

		Averaging Register <n>											
Bit	Weight	1	2	3	4	5	6	7	8	...	41	42	Bit is set to 1 when the following conditions exist:
0	1	2-42	3-42	4-42	5-42	6-42	7-42	8-42	9-42	...	42	--	Summary Bit - If any bit from that register fails, it propagates to the previous register, bit 0.
		Trace Numbers											
1	2	1	15	29	43	57	71	85	99	...	561	575	Averaging on this trace is complete
2	4	2	16	30	44	58	72	86	100	...	562	576	Averaging on this trace is complete
3	8	3	17	31	45	59	73	87	101	...	563	577	Averaging on this trace is complete
4	16	4	18	32	46	60	74	88	102	...	564	578	Averaging on this trace is complete

5	32	5	19	33	47	61	75	89	103	...	565	579	Averaging on this trace is complete
6	64	6	20	34	48	62	76	90	104	...	566	580	Averaging on this trace is complete
7	128	7	21	35	49	63	77	91	105	...	567	--	Averaging on this trace is complete
8	256	8	22	36	50	64	78	92	106	...	568	--	Averaging on this trace is complete
9	512	9	23	37	51	65	79	93	107	...	569	--	Averaging on this trace is complete
10	1024	10	24	38	52	66	80	94	108	...	570	--	Averaging on this trace is complete
11	2048	11	25	39	53	67	81	95	109	...	571	--	Averaging on this trace is complete
12	4096	12	26	40	54	68	82	96	110	...	572	--	Averaging on this trace is complete
13	8192	13	27	41	55	69	83	97	111	...	573	--	Averaging on this trace is complete
14	16384	14	28	42	56	70	84	98	112	...	574	--	Averaging on this trace is complete

To determine Register, Bit number, and Weight for trace numbers between 113 and 560 (not shown in the above table) use the following calculations.

The averaging status for trace numbers higher than 580 can NOT be tracked.

The following example calculates the Register, Bit number, and Bit Weight for trace # 400:

- To determine **Register** number, use $((\text{Trace \#} - 1) / 14) + 1$.
- To determine **Bit Number**, use the **remainder** +1 of the above calculation.
- $((400-1)/14) + 1 = \text{Register\# } r+1\text{Bit}$
 - $399/14 = 28 \text{ r}7$
 - $28+1= \text{Register } 29$
 - $7+1= \text{Bit number } 8$

- To determine **Bit Weight**: Use above table. For example: Bit 8 = 256

STATus:OPERation:DEFine<keyword>

Summarizes conditions in the OPERATION:Define:User<1|2|3> event registers.

<keyword>	Example
:CONDition?	STAT:OPER:DEF:COND?
:ENABle <bits>	STAT:OPER:DEF:ENAB 12
[:EVENTt]?	STAT:OPER:DEF?
:NTRansition <bits>	STAT:OPER:DEF:NTR 12
:PTRansition <bits>	STAT:OPER:DEF:PTR 0

Bit	Weight	Description	Bit is set to 1 when the following conditions exist:
1	2	USER1	any bit in the USER1 event register is set to 1
2	4	USER2	any bit in the USER2 event register is set to 1
3	8	USER3	any bit in the USER3 event register is set to 1

STATus:OPERation:DEFine:USER<1|2|3><keyword>

Monitors conditions that you define and map in any of the three OPER:DEF:USER event registers.

<keyword>	Example
:ENABle <bits>	STAT:OPER:DEF:USER1:ENAB 1024
[:EVENTt]?	STAT:OPER:DEF:USER1?
:MAP <bit>,<error>	STAT:OPER:DEF:USER1:MAP 0,-113 'when error -113 occurs, bit 0 in USER1 will set to 1.

Bit	Weight	Description	Bit is set to 1 when the following conditions exist:
0	1	for user	user defined
1	2	for user	user defined
2	4	for user	user defined
3	8	for user	user defined
4	16	for user	user defined
5	32	for user	user defined
6	64	for user	user defined
7	128	for user	user defined
8	256	for user	user defined
9	512	for user	user defined
10	1024	for user	user defined
11	2048	for user	user defined
12	4096	for user	user defined
13	8192	for user	user defined
14	16384	for user	user defined

STATus:OPERation:DEvice<keyword>

Summarizes conditions in the OPERation:DEvice event registers.

[<keyword>](#) **Example**

:CONDition? **STAT:OPER:DEV:COND?**

:ENABle <bits> **STAT:OPER:DEV:ENAB 16**

[:EVENTt]? **STAT:OPER:DEV?**

:NTRansition <bits> **STAT:OPER:DEV:NTR 16**

:PTRansition <bits> **STAT:OPER:DEV:PTR 0**

Bit	Weight	Description	Bit is set to 1 when the following conditions exist:
0	1	Unused	
1	2	Unused	
2	4	Unused	
3	8	Unused	
4	16	Sweep Completed	When sweep is complete
5	32	Unused	
6	64	Unused	
7	128	Unused	
8	256	Unused	
9	512	Unused	
10	1024	Unused	
11	2048	Unused	
12	4096	Unused	
13	8192	Unused	
14	16384	Unused	

Last modified:

13-Jan-2010 Changes to Limit and Average register
16-Jul-2009 Added note to Stat:Ques:Int:Meas register
16-Jul-2009 Fixed typo in STAT:QUES:INT register
9/19/06 MQ Modified for unlimited windows.

System Commands

Controls and queries settings that affect the PNA system.

```
SYSTem:
  ABORt:THReshold
  ACTive
    | CHANnel
    | MEASurement
  BEEPer:VOLume
  CAL:ALL More commands
  CAL:PHASe More commands
  CHANnels
    | CATalog?
    | COUPle[:STATe]
    | DELete
    | HOLD
    | RESume
  CLOCk[:STATe]
  COMMunicate More commands
  CONFigure
    | DIRectory
    | REVision
      | CPU?
      | DSP?
      | DSPFpga?
  CONFigure:EDEVice More commands
  CORRection
    | INTerpolate:LINear More commands
    | WIZard[:IMMediate ]
  ERRor?
    | COUNT?
    | REPort
      | SUNLeveled
  FIFO More commands
  FPReset
  MACRO:COPY
    | CHANnel[:TO]
    | SOURce
  MCLass
    | CATalog?
    | PARAmeter:CATalog?
  MEASurement
    | CATalog?
    | NAME?
    | TRACe?
    | WINDow?
```

POWer:| **LIMit**| **LOCK**| **STATe****PREFerences More commands****PRESet****SECurity**| **[LEVel]****SET****SHORtcut**| **ARGuments**| **DELete**| **EXECute**| **PATH**| **TITLe****TOUChscreen[:STATe]****UPReset**| **FPANel[:STATe]**| **LOAD[:FILE]**| **SAVE[:STATe]****WINDows**| **CATalog?**

Click on a [blue](#) keyword to view the command details.

See Also

- Referring to Traces Channels Windows and Meas Using SCPI
- Learn about PNA Preferences
- Example Programs
- Synchronizing the PNA and Controller
- SCPI Command Tree

SYSTem:ABORt:THReshold <value>

(Read-Write) When a PNA setting is made while a sweep is in progress, the sweep is immediately aborted by default. This command allows you to change that behavior by specifying a time threshold. When a setting change is made during a sweep and if the total sweep time is less than the threshold time, then the sweep is allowed to finish instead of immediately aborting.

For example, with a threshold setting of 60 seconds:

- Sweeps that require 60 seconds or less from start to finish will be allowed to complete if a PNA setting change is made at any time during the sweep.
- Sweeps that require MORE than 60 seconds from start to finish will be immediately aborted when a PNA setting change is made at any time during the sweep.

Notes:

- In general, PNA setting changes that could cause an aborted sweep are changes that affect how a measurement is made, such as changes in stimulus conditions.
- Sweep times are estimated.
- This setting affects ALL channels.

Parameters

<value> Threshold time in seconds. Set to 0 to immediately abort a sweep when a PNA setting is made.

Examples

```
SYST:ABOR:THR 10
```

```
'When a setting is made during a sweep, if that sweep requires less than 10 seconds more to complete, it will be allowed to finish instead of aborting.'
```

Query Syntax SYSTem:ABORT:THReshold?

Default 0 - No threshold time; all sweeps are immediately aborted.

SYSTem:ACTive:CHANnel?

(Read-only) Returns the number of the active channel. The active channel is the channel number that contains the active measurement. The active measurement is the trace that has a highlighted **Tr#** in the Trace Status area.

If there is no active channel, 0 is returned.

Examples

```
SYST:PRES
SYST:ACT:CHAN?
'Returns 1
```

Return Type Integer

Default Not Applicable

SYSTem:ACTive:MEASurement?

(Read-only) Returns the name of the active measurement. While looking at the PNA display, the active measurement is the trace that has a highlighted **Tr#** in the Trace Status area. Only displayed measurements can be active.

If there is no active measurement, " " (empty string) is returned.

Examples

```
SYST:PRES
SYST:ACT:MEAS?
'Returns "CH1_S11_1"
```

Return Type String

Default Not Applicable

SYSTem:BEEPer:VOLume <num>

(Read-Write) Sets and reads the volume of the internal speaker.

Parameters

<num> Relative volume of the internal speaker.

Choose a volume between 0 (off) and 100.

Examples

```
SYST:BEEP:VOL 5
system:beeper:volume
```

Query Syntax SYSTem:BEEPer:VOLume?

Default 0

SYSTem:CHANnels:CATalog?

(Read-only) Returns the channel numbers currently in use.

Examples

```
SYST:CHAN:CAT?  
  
system:channels:catalog?  
  
'Returns:  
  
"1,2,3"
```

Return Type String of comma-separated numbers

Default Not Applicable

SYSTem:CHANnels:COUPlE[:STATe] <bool>

(Read-Write) Sets and reads the state of channel coupling. This causes the PNA to emulate Agilent 8720 channel coupling.

When set to ON, all existing S-parameter channels receive the stimulus settings of the active channel. Subsequent changes made to any coupled channel are changed on all coupled channels.

Channels with applications such as SMC, VMC, GCA, Noise, IMD are not affected.

Coupling is primarily aimed at stimulus settings (such as start, stop, points, power) but also applies to many trigger settings and to Cal Set pointers.

Parameters

<bool> **ON** (or 1) Channels are coupled

OFF (or 0) Channels are NOT coupled

Examples

```
SYST:CHAN:COUP 1  
  
system:channels:couple:state OFF
```

Query Syntax SYSTem:CHANnels:COUPlE[:STATe]?

Default OFF

SYSTem:CHANnels:DELeTe <value>

(Write-only) Deletes the specified channel.

Parameters

<value> Channel number to delete

Examples `SYST:CHAN:DEL 2`

Query Syntax Not Applicable

Default Not Applicable

SYSTem:CHANnels:HOLD

(Write-only) Places all channels in hold mode. To place a single channel in hold mode, use SENS:SWE:MODE .

Examples `SYST:CHAN:HOLD`

Query Syntax Not Applicable

Default Not Applicable

SYSTem:CHANnels:RESume

(Write-only) Resumes the trigger mode of all channels that was in effect before sending SYSTem:CHANnels:HOLD (must be sent before SYST:CHAN:RESume).

Examples `SYST:CHAN:RES`

Query Syntax Not Applicable

Default Not Applicable

SYSTem:CLOCK[:STATe] <bool>

(Read-Write) Sets and reads the clock visibility state in the PNA status bar.

Parameters

<bool> **ON** (or 1) Clock is visible in the PNA status bar.

OFF (or 0) Clock is NOT visible in the PNA status bar.

Examples

```
SYST:CLOC 1
system:clock:state OFF
```

Query Syntax SYSTem:CLOCK[:STATe]?

Default ON

SYSTem:CONFigure <model>,<address>

(Write-only) Restarts as an "N-port" PNA using the specified multiport test set.

Learn more about PNA Multiport capability .

See other commands to configure multiport test sets.

Parameters

<model> String - Model of the test set with which to restart.

Use "Native" to restart without a test set.

To see a list of supported test sets, use SENS:MULT:CAT?

<address> Numeric - GPIB Address of the test set. Ignored when model = "Native".

Examples

```
SYST:CONF "NATIVE",0
system:configure "N44xx",18
```

Query Syntax Not Applicable

Default Not Applicable

SYSTem:CONFigure:DIRectory? <char>

(Read-only) Returns the directory path location for the specified file type.

Parameters None

<char> Type of file. Choose from:

STATE - This is the location for the storage of state files

- On Windows XP, this returns: "c:\program files\agilent\network analyzer\documents"
- On Windows 7, this returns: "c:\users\public\documents\network analyzer"

APPLication - This is the location of the PNA firmware executable files

- On Windows XP, this returns: "c:\program files\agilent\network analyzer"
- On Windows 7, this returns: "c:\program files (x86)\agilent\network analyzer"

SUPPort - This is the location of private support files for the PNA firmware

- On Windows XP, this returns: "c:\program files\agilent\network analyzer"
- On Windows 7, this returns: "c:\programdata\agilent\network analyzer"

Example `SYST:CONF:DIR? SUPP`

Return Type String

Default Not applicable

SYSTEM:CONFigure:REVision:CPU?

(Read-only) Returns a number that corresponds to the PNA CPU speed that is visible in the Help, About Network Analyzer dialog box. Learn more.

Use the following table to learn the clock speed using the returned value.

Reported CPU version - Clock speed

1.0 - 266 MHz

2.0 - 500 MHz

3.0 - 1100 MHz

4.0 - 1600 MHz

5.0 - 2000 MHz

6.0 - 2000 MHz dual core.

Parameters None

Example `SYST:CONF:REV:CPU?`

Return Type String

Default Not applicable

SYSTEM:CONFigure:REVision:DSP?

(Read-only) Returns the DSP Revision number that is visible in the Help, About Network Analyzer dialog box. Learn more.

Parameters None

Example `SYST:CONF:REV:DSP?`

Return Type String

Default Not applicable

SYSTEM:CONFigure:REVision:DSPFpga?

(Read-only) Returns the DSP FPGA Revision number that is visible in the Help, About Network Analyzer dialog box. Learn more.

Parameters None

Example `SYST:CONF:REV:DSPF?`

Return Type String

Default Not applicable

SYSTem:CORRection:WIZard[:IMMediate] <char>

(Write-only) Launches either the Calibration Wizard or the Version 2 Calibration Kit File Manager dialog box.

Remote operation returns immediately after the dialog is launched. This is done to avoid timeout issues with I/O protocols such as VISA. Although it is possible to send commands to the PNA while the dialog is open, this is not encouraged. Application programs should wait until the dialog is closed before resuming remote operations.

Parameters

<char> Choose from:

MAIN - Launches the Calibration Wizard which matches the current channel, such as standard S-params, NoiseFigure, GCA, and so forth.

CKIT - Launches the Version 2 Calibration Kit File Manager dialog box.

Both display on the PNA screen.

Examples `SYST:CORR:WIZ MAIN`
`system:correction:wizard:immediate ckit`

Query Syntax Not Applicable

Default Not Applicable

SYSTem:ERRor?

(Read-only) Returns the next error in the error queue. Each time the analyzer detects an error, it places a message in the error queue. When the `SYSTEM:ERROR?` query is sent, one message is moved from the error queue to the output queue so it can be read by the controller. Error messages are delivered to the output queue in the order they were received. The error queue is cleared when any of the following conditions occur:

- When the analyzer is switched ON.
- When the `*CLS` command is sent to the analyzer.
- When all of the errors are read.

If the error queue overflows, the last error is replaced with a "Queue Overflow" error . The oldest errors remain in the queue and the most recent error is discarded.

See list of all SCPI Errors.

Examples `SYST:ERR?`
`system:error?`

Default Not Applicable

SYSTEM:ERROR:COUNT?

(Read-only) Returns the number of errors in the error queue. Use `SYST:ERR?` to read an error.

See list of all SCPI Errors.

Examples `SYST:ERR:COUN?`
`system:error:count?`

Default Not Applicable

SYSTEM:ERROR:REPORT:SUNLeveled <bool>

(Read-Write) Specifies whether or not to report Source Unleveled errors to the SCPI system error buffer.

This setting will revert to the default (OFF) setting on Instrument Preset.

Parameters

<bool> **ON** (or 1) Report Source Unleveled Errors. Read errors from the system error buffer using SYST:ERR?

OFF (or 0) Do NOT report Source Unleveled Errors.

Examples

```
SYST:ERR:REP:SUNL 1
```

```
system:error:report:sunleveled ON
```

Query Syntax SYSTem:ERRor:REPort:UNLeveled?

Default OFF

SYSTem:FPRreset

(Write-only) Performs a standard Preset , then deletes the default trace, measurement, and window. The PNA screen becomes blank.

Examples

```
SYST:FPR  
system:fpreset
```

Default Not applicable

SYSTem:MACRo:COpy:CHANnel<cnum>[:TO] <num>

(Write-only) Copies ALL settings from <cnum> channel to <num> channel. Learn more about copy channels.

Use SENS:PATH:CONF:COPY to copy ONLY mechanical switch and attenuator settings.

Parameters

<cnum> Channel number to copy settings from. If unspecified, value is set to 1.

<num> Channel number to copy settings to.

Examples

```
SYST:MACR:COpy:CHAN1 2
```

```
system:macro:copy:channel12:to 3
```

Query Syntax Not Applicable

Default Not Applicable

SYSTem:MACRo:COpy:CHANnel<fromChan>:SOURce <fromPort>,<toChan>,<toPort>

(Write-only) Copies and applies an existing Source Power Calibration to another channel. Learn more about source power calibration .

Parameters

- <fromChan> Channel number of the existing source power correction.
- <fromPort> Port number of the existing source power correction.
- <toChan> Channel number to which the source power correction will be copied.
- <toPort> Port number to which the source power correction will be applied.

Examples

```
SYST:MACR:COpy:CHAN1:SOUR 1,2,1  
system:macro:copy:channel2:source 2,1,2
```

Query Syntax Not Applicable

Default Not Applicable

SYSTem:MCLass:CATalog?

(Read-only) Returns measurement classes available on the PNA. Learn more about Measurement Classes .

Parameters None

Examples

```
SYST:MCLass:CAT?
```

Return Type String of comma-separated measurement class names. See the complete list of measurement class names .

Default Not Applicable

SYSTem:MCLass:PARAmeter:CATalog? <name>

(Read-only) Returns ALL parameters that are supported by the specified measurement class.

Parameters

<name> String. Measurement Class name. See the complete list of measurement class names .

Examples

```
'Returns all parameters for Gain Compression.  
SYST:MCL:PAR:CAT? "Gain Compression"  
  
Return:  
  
"S11,S12,S13,S14,S21,S22,S23,S24,S31,S32,S33,S34,S41,S42,S43,S44,A,B,C,D,R,
```

Return Type String of comma-separated parameters

Default Not Applicable

SYSTem:MEASurement:CATalog? [chan]

(Read-only) Returns ALL measurement numbers, or measurement numbers from a specified channel.

Parameters

[chan] Optional. Channel number to catalog. If not specified, all measurement numbers are returned.

Examples

```
'Returns all measurement numbers  
SYST:MEAS:CAT?  
  
'Returns the measurement numbers on channel 2  
system:measurement:catalog? 2
```

Return Type String of comma-separated numbers

For example: "1,2"

Default Not Applicable

SYSTem:MEASurement<n>:NAME?

(Read-only) Returns the name of the specified measurement.

Parameters

<n> Measurement number for which to return the measurement name. If unspecified, value is set to 1.

Examples

```
'Returns the name of measurement 2  
SYST:MEAS2:NAME?
```

Return Type String

Default Not Applicable

SYSTEM:MEASurement<n>:TRACe?

(Read-only) Returns the trace number of the specified measurement number. Trace numbers restart for each window while measurement numbers are always unique.

Parameters

<n> Measurement number for which to return the trace number. If unspecified, value is set to 1.

Examples

```
'Returns the trace number of measurement 1  
SYST:MEAS1:TRAC?
```

Return Type Numeric

Default Not Applicable

SYSTEM:MEASurement<n>:WINDow?

(Read-only) Returns the window number of the specified measurement number.

Parameters

<n> Measurement number for which to return the window number. If unspecified, value is set to 1.

Examples

```
'Returns the window number of measurement 2  
SYST:MEAS2:WIND?
```

Return Type Numeric

Default Not Applicable

SYSTem:POWer<pnum>:LIMit <value>

(Read-Write) Sets and returns the power limit for the specified port. Learn more about Power Limit.

Parameters

<pnum> Port number. Choose any PNA port.

<value> Power limit in dBm

Examples

```
SYST:POW1:LIM 5
```

```
system:power2:limit 0
```

Query Syntax SYSTem:POWer<pnum>:LIMit?

Return Type Numeric

Default 100 dBm

SYSTem:POWer:LIMit:LOCK <bool>

(Read-Write) Enables or disables the ability to change the power limit values through the user interface. Learn more about Power Limit.

Parameters

<bool> Power limit lock. Choose from:

ON or 1 - Enables the ability to change the power limit values from the user interface.

OFF or 0 - Disables the ability to change the power limit values from the user interface.

Examples

```
SYST:POW:LIM:LOCK 1
```

```
system:power:limit:lock OFF
```

Query Syntax SYSTem:POWer:LIMit:LOCK?

Return Type Boolean

Default OFF

SYSTem:POWer<pnum>:LIMit:STATe <bool>

(Read-Write) Enables or disables the power limit for the specified port. Learn more about Power Limit.

Parameters

<pnum> Port number. Choose any PNA port.

<value> Power limit state. Choose from:

ON or 1 Enables the power limit for the port<pnum>.

OFF or 0 Disables the power limit for the port<pnum>.

Examples

```
SYST:POW1:LIM:STAT ON
system:power2:limit:state 0
```

Query Syntax SYSTem:POWer<pnum>:LIMit:STATe?

Return Type Boolean

Default OFF

SYSTem:PRESet

(Write-only) Deletes all traces, measurements, and windows. In addition, resets the analyzer to factory defined default settings and creates a S11 measurement named "CH1_S11_1". For a list of default settings, see Preset .

Regardless of the state of the User Preset Enable checkbox, the SYST:PRESet command will always preset the PNA to the factory preset settings, and SYST:UPReset will always perform a User Preset.

If the PNA display is disabled with DISP:ENAB OFF then SYST:PRES will NOT enable the display.

This command performs the same function as *RST with one exception: Syst:Preset does NOT reset Calc:FORMAT to ASCII as does *RST.

Examples

```
SYST:PRES
system:preset
```

Default Not applicable

SYSTem:SECurity[:LEVel] <char>

(Read-Write) Sets and returns the display of frequency information on the PNA screen and printouts.

Learn more about security level.

Parameters

<char> Choose from:

NONE - ALL frequency information is displayed.

LOW - NO frequency information is displayed. Frequency information can be redisplayed using the Security Setting dialog box or this command.

HIGH - LOW setting plus GPIB console is disabled. Frequency information can be redisplayed **ONLY** by performing a Preset, recalling an instrument state with None or Low security settings, or using this command.

EXTRa - HIGH setting plus:

- ASCII data saving is disabled. Same method to redisplay frequency information as HIGH setting.
- Mixer setup files (*.mxr) can NOT be saved.

Examples

```
SYST:SEC LOW
system:security:level high
```

Query Syntax SYSTem:SECurity[:LEVel]?

Return Type Character

Default None

SYSTem:SET <block>

(Read-Write) Sends a definite-length binary block Instrument state and sets the PNA with those settings. This command does the same as saving a *.sta file to the PNA (MMEM:STOR STATE) and then MMEM:TRAN to transfer the file to the computer.

Parameters

<block> The Instrument state file as definite-length arbitrary binary block.

Examples

```
SYST:SET <block>
```

Query Syntax SYSTem:SET? (This saves the instrument state file to the remote computer.)

Return Type Definite-length arbitrary binary block.

Default Not Applicable

SYSTem:SHORtcut<n>:ARGuments<string>

(Read-Write) Reads and writes the arguments for the specified macro. On the Edit Macro Dialog, this is called the "Macro run string parameters".

Parameters

<n> Numeric. Number of the macro that is stored in the PNA.

To find the number of a macro, either open the Macro Setup dialog and count the line number of the desired macro, or query the titles of all of the macros for the desired macro title.

<string> Arguments for the specified macro.

Examples

```
SYST:SHOR1:ARG  
"http://na.tm.agilent.com/pna/help/PNAWebHelp/help.htm"
```

Query Syntax SYSTem:SHORtcut<n>:ARGuments?

Default Not Applicable

SYSTem:SHORtcut<n>:DELeTe

(Write-only) Removes the specified macro from the list of macros in the PNA. Does not delete the macro executable file.

Parameters

<n> Numeric. Number of the macro that is stored in the PNA.

To find the number of a macro, either open the Macro Setup dialog and count the line number of the desired macro, or query the titles of all of the macros for the desired macro title.

Examples

```
SYST:SHOR1:DEL
```

Query Syntax Not Applicable

Default Not Applicable

SYSTem:SHORtcut<n>:EXECute

(Write-only) Executes (runs) the specified Macro (shortcut) that is stored in the PNA.

Parameters

<n> Numeric. Number of the macro that is stored in the PNA.

To find the number of a macro, either open the Macro Setup dialog and count the line number of the desired macro, or query the titles of all of the macros for the desired macro title.

Examples

```
SYST:SHOR1:EXEC
```

Query Syntax Not Applicable

Default Not Applicable

SYSTEM:SHORtcut<n>:PATH <string>

(Read-Write) Defines a Macro (shortcut) by linking a path and file name to the Macro number. To be executed, the executable file must be put in the PNA at the location indicated by this command.

Parameters

<n> Numeric. Number of the macro to be stored in the analyzer. If the index number already exists, the existing macro is replaced with the new macro.

<string> Full path, file name, and extension, of the existing macro "executable" file.

To find the number of a macro, either open the Macro Setup dialog and count the line number of the desired macro, or query the titles of all of the macros for the desired macro title.

Examples

```
SYST:SHOR1:PATH "C:/Program Files/Agilent/Network Analyzer/Documents/unguideMultiple.vbs"
```

Query Syntax SYSTEM:SHORtcut<n>:PATH?

Default Not Applicable

SYSTEM:SHORtcut<n>:TITLe<string>

(Read-Write) Reads and writes the name of the specified macro.

Parameters

<n> Numeric. Number of the macro that is stored in the PNA.

To find the number of a macro, either open the Macro Setup dialog and count the line number of the desired macro, or query the titles of all of the macros for the desired macro title.

<string> The name to be assigned to the macro.

Examples

```
SYST:SHOR1:TITL "Guided 4-Port Cal"
```

Query Syntax SYSTem:SHORtcut<n>:TITLe?

Default Not Applicable

SYSTem:TOUCHscreen[:STATe] <bool>

(Read-Write) Enables and disables the **PNA-X** touchscreen.

This setting remains until changed again from the front-panel or remotely, or until the hard drive is changed or reformatted.

Parameters

<bool> Choose from:

ON (1) Enables the touchscreen.

OFF (0) Disables the touchscreen.

Examples

```
SYST:TOUC 1  
system:touchscreen:state OFF
```

Query Syntax SYSTem:TOUCHscreen[:STATe]?

Return Type Boolean

Default ON when shipped from factory.

SYSTem:UPReset

(Write-only) Performs a User Preset. There must be an active User Preset state file (see Load and Save) or an error will be returned. Learn more about User Preset.

Regardless of the state of the User Preset Enable checkbox, the SYST:PRESet command will always preset the PNA to the factory preset settings, and SYST:UPReset will always perform a User Preset.

Examples `SYST:UPReset`
`system:upreset`

Query Syntax Not Applicable

Default Not Applicable

SYSTem:UPReset:FPANel[:STATe] <bool>

(Read-Write) 'Checks' and 'clears' the enable box on the User Preset dialog box . This only affects subsequent Presets from the front panel user interface.

Regardless of the state of the User Preset Enable checkbox, the SYST:PRESet command will always preset the PNA to the factory preset settings, and SYST:UPReset will always perform a User Preset.

Parameters

<bool> Front Panel User Preset State. Choose from:

0 User Preset OFF

1 User Preset ON

Examples `SYST:UPR:FPAN 1`
`system:upreset:fpanel:state 0`

Query Syntax SYSTem:UPREset:FPANel[:STATe]?

Return Type Boolean

Default 0

SYSTem:UPReset:LOAD[:FILE] <file>

(Write-only) Loads an existing instrument state file (.sta or .cst) to be used for User Preset. Subsequent execution of SYSTem:UPReset will cause the PNA to assume this instrument state.

Regardless of the state of the User Preset Enable checkbox, the SYST:PRESet command will always preset the PNA to the factory preset settings, and SYST:UPReset will always perform a User Preset.

Learn more about User Preset.

Parameters

<file> String - Name of the file to be loaded. The default folder "C:/Program Files/Agilent/Network Analyzer/Documents" is used if unspecified. Change the default folder name using MMEMory:CDIRectory .

Examples

```
SYST:UPR:LOAD '1MHzto20GHzUserPreset.cst'  
system:upreset:load:file 'C:/Documents and  
Settings/Administrator/My Documents/NewUserPreset.cst'
```

Query Syntax Not Applicable

Default Not Applicable

SYSTem:UPReset:SAVE[:STATE]

(Write-only) Saves the current instrument settings as UserPreset.sta. Subsequent execution of SYSTem:UPReset will cause the PNA to assume this instrument state.

Regardless of the state of the User Preset Enable checkbox, the SYST:PRESet command will always preset the PNA to the factory preset settings, and SYST:UPReset will always perform a User Preset.

Learn more about User Preset.

Examples

```
SYST:UPR:SAVE  
system:upreset:save:state
```

Query Syntax Not Applicable

Default Not Applicable

SYSTem:WINDows:CATalog?

(Read-only) Returns the window numbers that are currently being used.

Examples `SYST:WIND:CAT?`

`system:windows:catalog?`

Return Type String of comma-separated numbers.

For example: "1,2"

Default Not Applicable

Last modified:

7-Feb-2014	Changed Active commands to return 0 or "".
	Moved communicate commands.
15-Nov-2013	Fixed couple in tree
29-Apr-2013	Removed reference to 12 macros
6-Aug-2012	Fixed Meas:CAT and windows cat?
23-Mar-2012	Added Cal All
24-Jan-2012	Updated PSEnSOr command with 'any'.
5-Jan-2012	Added 6.0 CPU to list
26-Oct-2011	Added Abort command
11-Jan-2011	Minor edit
4-Nov-2010	Security for external sources (9.33)
14-Oct-2010	Added note to PMET:CAT?
16-Sep-2010	Added channel delete (A.09.30)
9-Apr-2010	Added Preset note to 'unleveled' command
30-Nov-2009	Added Help About read commands(9.1)
30-Jul-2009	Added syst:conf:edev and RTOF (9.0)
24-Feb-2009	Added Chan:Coup; Replaced True/False
4-Nov-2008	Added FIFO (8.33)
30-Oct-2008	Added several meas/trace/window query commands (8.33.x)
29-Sep-2008	Removed rev from PsenSOr example
17-Sep-2008	Added Syst:Pres vs *RST note
28-Aug-2008	Updated Launch Cal Wiz command
11-Feb-2008	Added Noise switch preference (8.2)
5-Feb-2007	Added Extra security and USB power meter commands
23-Feb-2007	Added touchscreen command
15-Nov-2006	Added Unleveled Error reporting
31-Oct-2006	Added PSRTrace command

SYSTem:CALibrate:ALL Commands

Contains the settings to configure a "Cal All" Calibration.

Use the [Guided Cal](#) interface to perform the calibration.

SYSTem:CALibrate:ALL:

CHANnel:

| [PORTs\[:SElect\]](#)

CSET:

| [CATalog?](#)

| [PREFix](#)

GUIDed:

| [CHANnel?](#)

| [PORTs?](#)

IFBW

MCLass:

| [PROPerTy:](#)

| [NAME:CATalog?](#)

| [VALue:](#)

| [CATalog?](#)

| [\[STATe\]](#)

PORT<n>:

| [RECeiver:ATTen](#)

| [SOURce:POWer:](#)

| [ATTen](#)

| [OFFSet](#)

| [\[VALue\]](#)

RESet

SElect

Click on a [blue](#) keyword to view the command details.

See Also

- [About Calibrate All Channels](#)

- [Example Program](#)
- [Guided Cal commands](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

SYSTem:CALibrate:ALL:CHANnel<ch>:PORTs[:SElect] <value>

(Write-Read) For each channel to be calibrated, sets and returns the ports to be calibrated. Specify port numbers ONLY for standard channels. [Application channels](#) are not necessary because they have designated input/output/LO ports.

Parameters

- <ch> Channel number to be calibrated.
- <value> Ports to be calibrated for the specified channel. Select any of the native PNA ports (1,2,3,4).

Examples

```
SYST:CAL:ALL:CHAN2:PORT 1,2,3
```

Query Syntax SYSTem:CALibrate:ALL:CHANnel<ch>:PORTs[:SElect]?

Return Type Comma-separated port numbers.

Default 1,2

SYSTem:CALibrate:ALL:CSET:CATalog?

(Read-only) Returns the User Cal Set or cal register names that were produced by the cal all session.

Parameters None

Examples

```
SYST:CAL:ALL:CSET:CATalog?
```

'returns this format:

```
"MyCalAll_STD_001, MyCalAll_SMC_002"
```

[See example program](#)

Return Type String of comma-separated Cal Set or cal register names

Default Not Applicable

SYSTem:CALibrate:ALL:CSET:PREFix<value>

(Write-Read) Sets and returns the prefix to be used when saving User Cal Sets that result from the Cal All session. The Meas Class and channel number are appended to this prefix for each calibrated channel. Use [SYST:CAL:ALL:CSET:CATalog?](#) to read the saved cal set names.

- [SENS:CORR:COLL:GUID:SAVE:CSET](#) can also be used to set the Cal Set prefix.
- If a Cal Set prefix is NOT set using either command, the cal data for each channel will be saved only to cal registers. [Learn about cal registers.](#)

Parameters

<value> (String) User Cal Set prefix.

Examples `SYST:CAL:ALL:CSET:PREFIX "MyCalAll"`

Query Syntax `SYSTEM:CALibrate:ALL:CSET:PREFIX?`

Return Type String

Default " " (Empty string)

SYSTEM:CALibrate:ALL:GUIDed:CHANnel?

(Read-only) Reads the channel number of the Cal All Calibration. Use this value as the <ch> argument for the subsequent [Guided:Cal](#) commands.

Parameters None

Examples `chan = SYST:CAL:ALL:GUID:CHAN?`

Return Type Numeric

Default Not applicable

SYSTEM:CALibrate:ALL:GUIDed:PORTs?

(Read-only) Returns the ports to be calibrated during the Cal All Channels calibration. Specify connectors and cal kits for these ports using the [Guided:Cal](#) commands.

Specify the ports to be calibrated for each channel using [SYST:CAL:ALL:CHAN<ch>:PORT](#).

Parameters None

Examples `ports = SYST:CAL:ALL:GUID:PORT?`

Return Type Comma-separated list of port numbers

Default Not applicable

SYSTem:CALibrate:ALL:IFBW <value>

(Write-Read) Sets and returns the IFBW for a Cal All calibration. [Learn more about this setting.](#)

Parameters

<value> IF Bandwidth in Hz. The list of valid IF Bandwidths is different depending on the PNA model. [See the list of valid settings.](#) If an invalid number is specified, the PNA will round up to the closest valid setting.

This command supports MIN and MAX as arguments. [Learn more.](#)

Examples

```
SYST:CAL:ALL:IFBW 10e3
```

Query Syntax SYSTem:CALibrate:ALL:IFBW?

Return Type Numeric

Default 1 kHz

SYSTem:CAL:ALL:MCLass:PROPerty:NAME:CATalog?

(Read-only) Returns the unique, settable properties for the current cal all session.

[See a list of valid properties and values for each measurement class.](#)

Parameters None

Examples

```
SYST:CAL:ALL:MCL:PROP:NAME:CAT?
```

'with NFX app, returns:

```
"Noise Cal Method,Noise Tuner,AutoOrient Tuner,Tuner In,Tuner Out,Receiver Characterization Method,ENR File,Noise Source Connector,Noise Source CalKit"
```

Return Type String of comma-separated properties.

Default Not applicable

SYSTem:CAL:ALL:MCLass:PROPerty:VALue:CATalog? <prop>

(Read-only) Returns the valid property values for a specific property name.

[See a list of valid properties and values for each measurement class.](#)

Parameters

<prop> (String) Property name for which valid values are to be returned.

Examples

```
SYST:CAL:ALL:MCL:PROP:VAL:CAT? "Noise Cal Method"

'with NFX app, returns:

"Scalar,Vector"
```

Return Type String of comma-separated values

Default Not applicable

SYSTem:CALibrate:ALL:MCLass:PROPerTy:VALue[:STATe] <prop>,<value>

(Write-Read) Sets and returns the property value for a specific property name.

[See a list of valid properties and values for each measurement class.](#)

Parameters

<prop> (String) Property name for which value is to be set or returned.

<value> Property value. To read a list of valid values, use [SYST:CAL:ALL:MCL:PROP:VAL:CAT?](#)

Examples

```
SYST:CAL:ALL:MCL:PROP:VAL "Noise Cal Method","Noise:Scalar"

SYST:CAL:ALL:MCL:PROP:VAL? "Noise Source Connector"
```

Query Syntax SYSTem:CALibrate:ALL:MCLass:PROPerTy:VALue[:STATe]? <prop>

Return Type String

Default Varies with the property name.

SYSTem:CALibrate:ALL:PATH:CONFig:ELEMent[:STATe] <element>,<setting>

(Write-Read) Sets and returns the Path Configuration settings for a Cal All calibration. [Learn more about this setting.](#)

Parameters

<element> (String) Path configuration element to be set. [See a list of configurable RF Path elements and settings.](#)

<setting> (String) Path configuration element setting.

Examples `SYST:CAL:ALL:PATH:CONFig:ELEMent "Port1NoiseTuner","Internal"`

Query Syntax `SYSTem:CALibrate:ALL:PATH:CONFig:ELEMent[:STATe]? <element>`

Return Type String

Default Not Applicable

SYSTem:CALibrate:ALL:PORT<n>:RECeiver:ATTen<value>[,src]

(Write-Read) Sets and returns the Receiver Attenuator setting for a Cal All calibration.

Parameters

<n> Receiver port number.

<value> Attenuation value in dB for a Cal All calibration. Choose a valid value for the PNA model. [See valid settings.](#)

[src] String. (NOT case sensitive). Source port. Optional. Use [SOUR:CAT?](#) to return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, [true mode balanced port](#), or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples `SYST:CAL:ALL:PORT2:REC:ATT 10`

Query Syntax `SYSTem:CALibrate:ALL:PORT<n>:RECeiver:ATTen?`

Return Type Numeric

Default 0

SYSTem:CALibrate:ALL:PORT<n>:SOURce:POWER:ATTen<value>[,src]

(Write-Read) Sets and returns the Source Attenuator setting for the Cal All calibration.

Parameters

- <n> Source port number.
- <value> Attenuation value in dB for the Cal All calibration. Choose a valid value for the PNA model. [See valid settings](#).
- [src] String. (NOT case sensitive). Source port. Optional. Use [SOUR:CAT?](#) to return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, [true mode balanced port](#), or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

```
SYST:CAL:ALL:PORT2:SOUR:POW:ATT 10
```

Query Syntax SYSTem:CALibrate:ALL:PORT<n>:SOURce:POWer:ATTen?

Return Type Numeric

Default 0

SYSTem:CALibrate:ALL:PORT<n>:SOURce:POWer:OFFSet <value>[,src]

(Write-Read) Sets and returns the power offset value for a Cal All calibration.

Power Offset provides a method of compensating port power for added attenuation or amplification in the source path. The result is that power at the specified port reflects the added components.

Parameters

- <n> Source port number.
- <value> Power offset value in dB for a Cal All calibration.
- For amplification, use positive offset.
 - For attenuation, use negative offset.
- [src] String. (NOT case sensitive). Source port. Optional. Use [SOUR:CAT?](#) to return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, [true mode balanced port](#), or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1

Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples `SYST:CAL:ALL:PORT2:SOUR:POW:OFFS 10`

Query Syntax `SYSTem:CALibrate:ALL:PORT<n>:SOURce:POWer:OFFSet?`

Return Type Numeric

Default 0

SYSTem:CALibrate:ALL:PORT<n>:SOURce:POWer[:VALue] <value>[,src]

(Write-Read) Sets and returns the power level at which a Cal All calibration is to be performed.

Parameters

<n> Source port number.

<value> Power level at which the calibration is to be performed.

[src] String. (NOT case sensitive). Source port. Optional. Use [SOUR:CAT?](#) to return a list of valid port names.

While this argument can be used to make settings for ALL ports, it is designed to access ports such as an external source, [true mode balanced port](#), or one of the Source 2 outputs on the 2-port 2-source PNA-X model such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples `SYST:CAL:ALL:PORT2:SOUR:POW 0`

Query Syntax `SYSTem:CALibrate:ALL:PORT<n>:SOURce:POWer[:VALue]?`

Return Type Numeric

Default Preset power of the PNA model.

[See the data sheet for the power level for each model.](#)

SYSTem:CALibrate:ALL:RESet

(Write-only) Resets all properties associated with the Cal All session to their default values.

Parameters None

Examples `SYST:CAL:ALL:RES`

Query Syntax Not Applicable

Default Not Applicable

SYSTem:CALibrate:ALL:SElect <value>

(Write-Read) Sets and returns the list of channels to be calibrated during the Cal All session.

Parameters

<value> Channel numbers to be calibrated. These channels must already exist.

Examples

```
SYST:CAL:ALL:SEL 1,2,3
```

Query Syntax SYSTem:CALibrate:ALL:SElect?

Return Type Comma-separated channel numbers.

Default Existing channels

Last modified:

10-Feb-2014	Added links to properties and values
19-Dec-2013	Added optional source port strings
19-Sep-2012	Removed LO cal and minor edits (JK)
5-Jan-2012	New topic

System:Calibrate:Phase

Contains the settings to perform an SMC Phase Reference Calibration.

SYSTem:CALibrate:PHASe

[CKIT](#)

[CONNector](#)

[DEEMbed](#)

FREQuency:

| [STARt](#)

| [STOP](#)

GUIDed:

| [CHANnel?](#)

[PORT](#)

[POWER:ATTenuator](#)

REFerence:

| [CATalog?](#)

[RESet](#)

UNKNown:

| [INCLude](#)

| [INPut:POWER](#)

| [LO](#)

| [FREQuency](#)

| [POWER](#)

Click on a [blue](#) keyword to view the command details.

Important Notes

- It is NOT necessary to create an SMC measurement before performing a **remote** Phase Reference Cal. It is necessary when performed from the user interface.
- Before A..09.90, port selection was made remotely by selecting connectors and Cal Kits for the ports to be included in the SOLT calibration. With A.09.90, port selection is made explicitly with the commands in this node.

See Also

- [Example Program](#)
 - [About Phase Reference Cal](#)
 - [Guided Cal commands](#)
 - [Synchronizing the PNA and Controller](#)
 - [SCPI Command Tree](#)
-

SYSTem:CALibrate:PHASe:CKIT <string>

(Write-Read) Sets and returns the Cal Kit that will be used to perform the S-parameter Cal.

To read a list of valid Cal Kits, use [SENSe:CORR:COLL:GUID:CKIT:CAT?](#)

Parameters

<string> Cal Kit.

Examples

```
SYST:CAL:PHAS:CKIT "85052D"
```

Query Syntax

```
SYSTem:CALibrate:PHASe:CKIT?
```

Return Type

String

Default

" "

SYSTem:CALibrate:PHASe:CONNector <string>

(Write-Read) Sets and returns the connector type and gender of your Cal Kit.

To read a list of valid connector types, use [SENS:CORR:COLL:GUID:CONN:CAT?](#)

Parameters

<string> Connector type.

Examples

```
SYST:CAL:PHAS:CONN "APC 3.5 female"
```

Query Syntax

```
SYSTem:CALibrate:PHASe:CONNector?
```

Return Type

String

Default

" "

SYSTem:CALibrate:PHASe:DEEMbed <bool>

(Write-Read) Sets and returns the state of de-embedding (reversing) the port 2 coupler.

Parameters

<bool> Port 2 coupler de-embed state. Choose from:

ON (or **1**) - Configures the calibration to include additional measurements to de-embed the effects of reversing the coupler. (This is the same as clearing the “Omit Coupler” checkbox.)

OFF (or **0**) - Excludes additional measurements for de-embedding the effects of reversing the coupler.

Examples

```
SYST:CAL:PHAS:DEEM 1
```

Query Syntax SYSTem:CALibrate:PHASe:DEEMbed?

Return Type Boolean

Default ON or 1

SYSTem:CALibrate:PHASe:FREQuency:STARt <value>

(Write-Read) Sets and returns the phase reference cal start frequency.

Parameters

<value> Start frequency. Choose any frequency from 10 MHz to the stop frequency of the PNA.

Examples

```
SYST:CAL:PHAS:FREQ:STAR 17.5e6
```

Query Syntax SYSTem:CALibrate:PHASe:FREQuency:STARt?

Return Type Numeric

Default Start frequency of the PNA

SYSTem:CALibrate:PHASe:FREQuency:STOP <value>

(Write-Read) Sets and returns the phase reference cal stop frequency.

Parameters

<value> Stop frequency. Choose any frequency within the range of the PNA.

Examples

```
SYST:CAL:PHAS:FREQ:STOP 26.5e9
```

Query Syntax SYSTem:CALibrate:PHASe:FREQuency:STOP?

Return Type Numeric

Default Stop frequency of the PNA

SYSTem:CALibrate:PHASe:GUIDed:CHANnel?

(Read-only) Reads the channel number of the Phase Reference Calibration. Use this value as the <ch> argument for the subsequent [Guided:Cal](#) commands.

Parameters None

Examples

```
chan = SYST:CAL:PHAS:GUID:CHAN?
```

Return Type Numeric

Default Not applicable

SYSTem:CALibrate:PHASe:PORT<n> <bool>

(Write-Read) Sets and returns the enable state for the specified port.

Parameters

<n> Port number to enable or disable.

<bool> Port enable state. Choose from:

ON (or **1**) - Enable port <n>

OFF (or **0**) - Disable port <n>

Examples

```
SYST:CAL:PHAS:PORT2 1
```

Query Syntax SYSTem:CALibrate:PHASe:PORT<n>?

Return Type Boolean

Default Ports 1 and 2 are enabled.

Ports 3 and 4 (if present) are disabled

SYSTem:CALibrate:PHASe:POWER:ATTenuator <value>

(Write-Read) Sets and returns the Source Attenuator setting for the Phase Reference calibration.

Note: This setting MUST match the source attenuator setting at the mixer input port for subsequent SMC+Phase measurements.

Parameters

<value> Attenuation value in dB. Choose a valid value for the PNA model. [See valid settings.](#)

Examples

```
SYST:CAL:PHAS:POW:ATT 10
```

Query Syntax SYSTem:CALibrate:PHASe:POWER:ATTenuator?

Return Type Numeric

Default 10 dB

SYSTem:CALibrate:PHASe:REFerence <string>

(Write-Read) Sets and returns the Phase Reference ID to be used for the Phase Reference calibration. Use SYST:CAL:PHAS:REF:CAT? to read the phase references currently connected to the PNA USB.

Parameters

<string> Phase reference ID string.

Examples

```
SYST:CAL:PHAS:REF "MYPRT0001"
```

Query Syntax SYSTem:CALibrate:PHASe:REFerence?

Return Type String

Default Not Applicable

SYSTem:CALibrate:PHASe:REFerence:CATalog?

(Read-only) Reads the ID strings of the phase references that are currently connected to the PNA USB.

Parameters None

Examples

```
pRef = SYST:CAL:PHAS:REF:CAT?
```

Return Type Comma-separated string

Default Not Applicable

SYSTem:CALibrate:PHASe:RESet

(Write-only) Resets all properties associated with the Phase Reference Cal to their default values.

Parameters None

Examples SYST:CAL:ALL:PHAS:RES

Query Syntax Not Applicable

Default Not Applicable

SYSTem:CALibrate:PHASe:UNKNown:INCLude <bool>

(Write-Read) Sets and returns the state of Unknown Mixer calibration.

Parameters

<bool> Unknown Mixer cal state. Choose from:

ON (or **1**) - Enable Unknown Mixer cal. The start frequency becomes 10 MHz and can NOT be changed.

OFF (or **0**) - Disable Unknown Mixer cal.

Examples SYST:CAL:PHAS:UNKN:INCL 1

Query Syntax SYSTem:CALibrate:PHASe:UNKNown:INCLude?

Return Type Boolean

Default OFF or 0

SYSTem:CALibrate:PHASe:UNKNown:INPut:POWer <value>

(Write-Read) Sets and returns the input power level to the unknown mixer.

Parameters

<value> Input power level in dBm.

Examples SYST:CAL:PHAS:UNKN:INP:POW -5

Query Syntax SYSTem:CALibrate:PHASe:UNKNown:INPut:POWer?

Return Type Numeric

Default -15 dBm

SYSTem:CALibrate:PHASe:UNKNown:LO:FREQUency <value>

(Write-Read) Sets and returns the LO frequency to the unknown mixer.

Parameters

<value> LO frequency in Hz. Choose a value between 3 GHz and (Max Frequency minus 1GHz).

For a 26.5 GHz PNA, the range is 3 GHz to 25.5 GHz.

For best results, use the default LO frequency. 3.351Ghz. This frequency produces no spurs from the input/LO frequency. And also the Input frequency will have no band breaks.

Examples

```
SYST:CAL:PHAS:UNKN:LO:FREQ 3.351e9
```

Query Syntax SYSTem:CALibrate:PHASe:UNKNown:LO:FREQuency?

Return Type Numeric

Default 3.351 GHz

SYSTem:CALibrate:PHASe:UNKNown:LO:POWer <value>

(Write-Read) Sets and returns the LO power level to the unknown mixer.

Parameters

<value> LO power level in dBm.

Examples

```
SYST:CAL:PHAS:UNKN:LO:POW 10
```

Query Syntax SYSTem:CALibrate:PHASe:UNKNown:LO:POWer?

Return Type Numeric

Default 10 dBm

Last modified:

- 13-Feb-2013 Added UNKN commands (9.90)
- 14-Aug-2012 Fixed link to example
- 3-Apr-2012 New topic

External Device Commands

Configures and makes settings for an external device.

SYST:CONF:EDEvice:

| [ADD](#)

| [CAT?](#)

| [DRIVer](#)

| [DTYPe](#)

| [EXISts?](#)

| [IOConfig](#)

| [IOENable](#)

| [LOAD](#)

| [REMove](#)

| [SAVE](#)

| [STATe](#)

| [TOUT](#)

| [DC More commands](#)

| [PMAR More commands](#)

| [PULSe More commands](#)

| **SOURce:**

| [DPP](#)

| [TMODe](#)

| [TPORt](#)

Click on a [blue](#) keyword to view the command details.

See Also

- Learn about: [Configure an External Source](#)
- Learn about: [Configure a PMAR Device](#)
- Example: [Configure an External Source](#)
- Example: [Configure a PMAR Device](#)
- [SYST:PREF:ITEM:EDEV:DPOL](#) - Determines whether external devices remain activated or are de-activated when the PNA is Preset or when a Instrument State is recalled.
- [Synchronizing the PNA and Controller](#)

- [SCPI Command Tree](#)
-

SYSTem:CONFigure:EDEvice:ADD <name>

(Write-only) Adds an external device to the list of configured devices. This is the same as pressing **New** on the [Select an external device](#) dialog.

Upon creation, all settings on the new device are set to the defaults. The device is not active until set using [SYST:CONF:EDEV:STAT](#)

Parameters

<name> String - Model and type of the external device.

To see a list of configured external devices, use [SYST:CONF:EDEV:CAT?](#)

Examples

```
SYST:CONF:EDEV:ADD "myDevice"  
system:configure:edev:add "myDevice"
```

Query Syntax Not Applicable

Default Not Applicable

SYSTem:CONFigure:EDEvice:CAT?

(Read-only) Returns a list of names of all configured devices. These are devices that appear in the [external devices](#) dialog.

Use [SENS:FOM:CAT?](#) to report all **active** devices.

Use [Source:CAT?](#) to report all **active** sources.

Parameters None

Example

```
SYST:CONF:EDEV:CAT?  
system:configure:edev:cat
```

Return Type String of comma-separated devices. "Device0:Driver0, Device1:Driver1"

Default Not applicable

SYSTem:CONFigure:EDEvice:DRIVER <name>,<value>

(Read-Write) Sets and returns the external device driver (model).

Parameters

- <name> String - Name of the device.
 - <value> String - External device driver (model). Choose from the following:
 - AGPM** for all power meters.
 - AGPULSEGEN** for supported pulse generators.
 - DCSource** for all supported DC Sources
 - DCMeter** for all supported DC Meters
- [See a list of supported external source drivers.](#)

Examples

```
SYST:CONF:EDEV:DRIV "myDevice","AGPM"  
system:configure:edev:driver "myDevice","AGESG"
```

Query Syntax SYSTem:CONFigure:EDEVice:DRIVER? <name>

Return Type String

Default "AGGeneric"

SYSTem:CONFigure:EDEVice:DTYPE <name>,<type>

(Read-Write) Sets and returns the Device Type for the external device.

Parameters

- <name> String - Name of the device to modify.
- <type> String - Device type - not case sensitive. Choose from:
 - "Source"** - external source
 - "Power Meter"** - power meter
 - "DC Meter"** - DC voltmeter
 - "DC Source"** - DC power supply
 - "Pulse Generator"** - external pulse generator

Examples

```
SYST:CONF:EDEV:DTYP "myDevice","Power Meter"  
system:configure:edev:dtype "myDevice","Source"
```

Query Syntax SYSTem:CONFigure:EDEVice:DTYPE? <name>

Return Type String

Default None

SYSTem:CONFigure:EDEVice:EXISts? <string>

(Read-only) Returns whether the named device is present on the bus for which it is configured.

Parameters

<string> Name of the external device.

Example `SYST:CONF:EDEV:EXIS? "MyPowerMeter"`

Return Type Boolean

- **0** - The device is not in the collection or the device fails to respond and times out when communication is attempted.
- **1** - The device responds when communication is attempted.

Default Not applicable

SYSTem:CONFigure:EDEVice:IOConfig <name>,<value>

(Read-Write) Sets and return the configuration path for the specified external device.

Parameters

<name> String - Name of the device.

<value> String - Configuration path. Any valid VISA resource shown in the IO Configuration field of the [external devices dialog](#), enclosed in quotes.

Do NOT use the ID string of a PMAR USB power sensor as the resource string. The ID string is returned by [SYST:COMM:USB:PMET:CAT?](#)

Examples `SYST:CONF:EDEV:IOC "myDevice","GPIB0::13::INSTR"`
`system:configure:edev:ioconfig "myDevice","GPIB0::13::INSTR"`

Query Syntax SYSTem:CONFigure:EDEVice:IOConfig? <name>

Return Type String

Default " " Empty String

SYSTem:CONFigure:EDEvice:IOENable <name>,<value>

(Read-Write) Enable or disable communication with an external device.

When disabled (OFF), the PNA will NOT attempt to connect to the external device regardless of the instrument state command ([SYST:CONF:EDEV:STATe](#)). Therefore, no errors will be produced if the device is not connected.

This command is useful for debugging and testing states when the external device is not connected. This command is unnecessary in ordinary operation (when the device is connected).

Parameters

- <name> String - Name of the device.
- <value> Boolean - Choose from:
- OFF** or **0** - Device communication disabled
 - ON** or **1** - Device communication enabled

Examples

```
SYST:CONF:EDEV:IOEN "myDevice", ON  
system:configure:edev:ioenable "myDevice", 0
```

Query Syntax SYSTem:CONFigure:EDEvice:IOENable? <name>

Return Type Boolean

Default ON

SYSTem:CONFigure:EDEvice:LOAD <file>,<name>

(Write-only) Recalls an external device configuration file from the PNA hard drive.

Currently, only DC Supply and DC Meter configuration files are supported. See more [DC Device commands](#).

Use [SYST:CONF:EDEV:SAVE](#) to save a configuration file.

Parameters

- <file> String - Filename of the external device configuration file.
- <name> String - Name of the external device. Currently, only DC Supply and DC Meter configuration files are supported.

Examples

```
SYST:CONF:EDEV:LOAD "myDevice.xml", "MyDCMeter"
```

Query Syntax Not Applicable

Default Not Applicable

SYSTem:CONFigure:EDEvice:REMOve <name>

(Write-only) Removes the specified device from the list of configured devices. If the device is a Source and both Active and I/O Enabled is checked (ON), then the RF power state is set to OFF.

[Learn more.](#)

Parameters

<name> String - Name of the device. Not case sensitive. Use [SYST:CONF:EDEV:CAT?](#) to return a list of configured devices.

Examples

```
SYST:CONF:EDEV:REM "myDevice"  
system:configure:edev:remove "myDevice"
```

Query Syntax Not Applicable

Default Not Applicable

SYSTem:CONFigure:EDEvice:SAVE <file>,<name>

(Write-only) Saves an external device configuration file to the PNA hard drive.

Currently, only DC Supply and DC Meter configuration files are supported. See more [DC Device commands](#).

Use [SYST:CONF:EDEV:LOAD](#) to recall a configuration file.

Parameters

<file> String - Filename of the external device configuration file.

<name> String - Name of the external device. Currently, only DC Supply and DC Meter configuration files are supported.

Examples

```
SYST:CONF:EDEV:SAVE "myDevice.xml", "MyDCSupply"
```

Query Syntax Not Applicable

Default Not Applicable

SYSTem:CONFigure:EDEvice:STATe <name>,<value>

(Read-Write) Set and return the state of activation of the device. When [SYST:CONF:EDEV:IOEN](#) = ON, and this command is set to ON, the PNA will attempt communication with the external device. An error is returned if communication cannot be verified.

Send this command AFTER sending other external device settings (especially [SYST:CONF:EDEV:DTYP](#)) to avoid communicating with the device before it has been fully configured.

See Also: [SYST:PREF:ITEM:EDEV:DPOL](#) - Determines whether external devices remain activated or are deactivated when the PNA is Preset or when a Instrument State is recalled.

Parameters

- <name> String - Name of the device.
- <value> Boolean - Choose from:
 - OFF** or **0** - Device is NOT activated
 - ON** or **1** - Device is activated.

Examples

```
SYST:CONF:EDEV:STAT "myDevice", ON  
system:configure:edev:state "myDevice", 0
```

Query Syntax SYSTem:CONFigure:EDEVice:STATe? <name>

Return Type Boolean

Default OFF - When configured using the front panel user interface, the device is ON (activated) by default.

SYSTem:CONFigure:EDEVice:TOUT <name>,<value>

(Read-Write) Set and return the time out value for the specified external device. This is the time allowed for communication with the device before an error is generated.

Parameters

<name> String - Name of the device.

<value> Time out value in seconds.

Examples

```
SYST:CONF:EDEV:TOUT "myDevice",2
system:configure:edevicetout "myDevice",5
```

Query Syntax SYSTem:CONFigure:EDEVice:TOUT? <name>

Return Type Numeric

Default 20

SYSTem:CONFigure:EDEVice:SOURce:DPP <name>,<value>

(Read-Write) Sets and returns the amount of time the PNA should wait after for an external source to settle before making a measurement at each data point. This setting applies to all channels that use this external source.

Parameters

<name> String - Name of the device.

<value> Dwell time in seconds.

Examples

```
SYST:CONF:EDEV:SOUR:DPP "myDevice",2
system:configure:edevicetout "myDevice",.1
```

Query Syntax SYSTem:CONFigure:EDEVice:SOURce:DPP? <name>

Return Type Numeric

Default 3.114 e-3

SYSTem:CONFigure:EDEVice:SOURce:TMODe <name>,<value>

(Read-Write) Sets and returns the trigger mode for an external source. [Learn more.](#)

Parameters

<name> String - Name of the device.

<value> Trigger Mode. Choose from:

CW - Software CW mode

HW - Hardware list mode

Examples

```
SYST:CONF:EDEV:SOUR:TMOD "myDevice",CW
```

```
system:configure:edev:source:tmode "myDevice",hw
```

Query Syntax SYSTem:CONFIgure:EDEVice:SOURce:TMODe? "myDevice"

Return Type Character

Default Depends on Source and PNA Model

SYSTem:CONFIgure:EDEVice:SOURce:TPORt <name>,<value>

(Read-Write) Sets and returns the PNA port through which an external source is to be triggered.

Parameters

<name> String - Name of the device.

<value> Trigger Port. Choose from **aux1** or **aux2**

Examples

```
system:configure:edev:source:tport "myDevice",aux1
```

Query Syntax SYSTem:CONFIgure:EDEVice:SOURce:TPORt? <name>

Return Type Character

Default **aux1**

Last Modified:

17-Sep-2012	Edited IOEnable and State commands
30-Aug-2012	Fixed three typos (MM)
11-Apr-2012	Edited Save and Load commands
14-Oct-2010	Added note to IOConfig
24-Mar-2010	Fixed configure and added Remove
15-Jan-2010	Added "AG" to driver example
8-Dec-2009	Changed IOEnable default state and other edits
28-Mar-2009	MX New topic (9.0)

SYSTem:CONF:EDEvice:DC Commands

Configures external SMU, DC Meter, and DC Source properties.

SYST:CONF:EDEvice:DC

| [CORRection](#)

| [DPOint](#)

| [DSWEEP](#)

| [OFFSet](#)

| [SCALe](#)

| [TYPE](#)

Click on a [blue](#) keyword to view the command details.

See Also

- All [SYST:CONF:EDEV](#) commands
- [SOURce:DC](#) commands (make DC sweep settings)
- Learn about: [Configure an External DC Device](#)
- Learn about [Configure an External Device](#)
- [SYST:PREF:ITEM:EDEV:DPOL](#) - Determines whether External Devices remain activated or are de-activated when the PNA is Preset or when a Instrument State is recalled.
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

SYSTem:CONFigure:EDEvice:DC:CORRection <name>,<value>

(Read-Write) Sets and returns the correction ON/OFF state for a DC Meter and a DC Source.

Parameters

- <name> String - Name of the device.
- <value> Correction ON/OFF state. Choose from:

ON or 1 - Turn Correction ON

OFF or 0 - Turn Correction OFF

Examples

```
SYST:CONF:EDEV:DC:CORR "myDCDevice",1
system:configure:edev:dc:correction "myDCDevice",OFF
```

Query Syntax SYSTem:CONFigure:EDEVice:DC:CORRection? <name>

Return Type Boolean

Default OFF

SYSTem:CONFigure:EDEVice:DC:DPOint <name>,<value>

(Read-Write) Sets and returns the "Dwell Before/After Point" value for an external DC Device which can be configured as either a DC Meter or a DC Source.

Parameters

- <name> String - Name of the device.
- <value> For DC Meter, the dwell time (in seconds) before making a data point measurement.
For DC Source, the dwell time (in seconds) after making a data point setting.

Examples

```
SYST:CONF:EDEV:DC:DPO "myDCDevice",10e-3
system:configure:edev:dc:dpoint "myDCDevice",.01
```

Query Syntax SYSTem:CONFigure:EDEVice:DC:DPOint? <name>

Return Type Numeric

Default 3 milliseconds

SYSTem:CONFigure:EDEVice:DC:DSWeep <name>,<value>

(Read-Write) Sets and returns the "Dwell Before Sweep" value for an external DC Device which can be configured as either a DC Meter or a DC Source.

Parameters

- <name> String - Name of the device.
- <value> The dwell time (in seconds) before making a new sweep.

Examples

```
SYST:CONF:EDEV:DC:DSW "myDCDevice",10e-3  
system:configure:edev:dc:dswEEP "myDCDevice",.01
```

Query Syntax SYSTem:CONFigure:EDEVice:DC:DSWEEP? <name>

Return Type Numeric

Default 1 millisecond

SYSTem:CONFigure:EDEVice:DC:OFFSet <name>,<value>

(Read-Write) Sets and returns the offset correction value for an external DC Device which can be configured as either a DC Meter or a DC Source.

Parameters

- <name> String - Name of the device.
- <value> DC offset value.

The PNA will display readings from a DC Meter as:

```
Display = (Meas'd value - Offset) * Scale
```

The PNA will adjust the output from a DC Source as:

```
Output = (Set value - Offset) * Scale
```

Examples

```
SYST:CONF:EDEV:DC:OFFS "myDCDevice",4  
system:configure:edev:dc:offset "myDCDevice",1.25
```

Query Syntax SYSTem:CONFigure:EDEVice:DC:OFFSet? <name>

Return Type Numeric

Default 0

SYSTem:CONFigure:EDEVice:DC:SCALE <name>,<value>

(Read-Write) Sets and returns the scale correction value for an external DC Device which can be configured as either a DC Meter or a DC Source.

Parameters

<name> String - Name of the device.

<value> DC Scale value.

The PNA will display readings from a DC Meter as:

```
Display = (Meas'd value - Offset) * Scale
```

The PNA will adjust the output from a DC Source as:

```
Output = (Set value - Offset) * Scale
```

Examples

```
SYSTem:CONF:EDEV:DC:SCAL "myDCDevice",1.2
```

```
system:configure:edev:dc:scale "myDCDevice",.5
```

Query Syntax SYSTem:CONF:figure:EDEV:dc:SCALE? <name>

Return Type Numeric

Default 1

SYSTem:CONF:figure:EDEV:dc:TYPE <name>,<value>

(Read-Write) Sets and returns the DC Type for an external DC Device which can be configured as either a DC Meter or a DC Source. This setting is used as the units for display on the PNA X-axis.

Parameters

<name> String - Name of the device.

<value> DC type. Choose from:

"dBm", "A", "V", "W", "K", "F", "C"

Examples

```
SYSTem:CONF:EDEV:DC:TYPE "myDCDevice","A"
```

```
system:configure:edev:dc:type "myDCDevice","w"
```

Query Syntax SYSTem:CONF:figure:EDEV:dc:TYPE? <name>

Return Type String

Default "V"

Last modified:

30-Oct-2013 Removed unimplemented commands
 Added commands for SMU
 New topic

SYST:CONF:EDEvice:PMAR Commands

Configures and makes settings for an external Power Meter as Receiver.

SYSTem:CONFigure:EDEVice:PMAR

- | [CALibrate](#)
- | [FLIMit](#)
- | [FMAXimum](#)
- | [FMINimum](#)
- | [READing:](#)
 - | [COUNT](#)
 - | [NTOLerance](#)
- | [SENSor](#)
- | [TABLe:](#)
 - | [CFAC:](#)
 - | [DATA](#)
 - | [FREQuency](#)
 - | [LOSS:](#)
 - | [DATA](#)
 - | [FREQuency](#)
 - | [STATe](#)
 - | [RFACTor](#)
- | [ZERO](#)

Click on a [blue](#) keyword to view the command details.

See Also

- Learn about: [Configure a Power Meter As Receiver](#)
- See root [SYST:CONF:EDEV](#) commands
- Learn about [Configure and External Device](#)
- [SYST:PREF:ITEM:EDEV:DPOL](#) - Determines whether External Devices remain activated or are de-activated when the PNA is Preset or when a Instrument State is recalled.
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

SYSTem:CONFigure:EDEVice:PMAR:CALibrate <name>

(Write-only) Performs a calibration of the power sensor. Calibration usually involves connecting the power sensor to the meter's 1 mW reference.

- Agilent P-Series sensors have an internal reference so you can calibrate them using this command without connecting to the meters reference port.
- Agilent USB power sensors do not require calibrating.
- For other sensors, refer to the documentation to determine if it has calibration capability.

This command is always synchronous, so *OPC? is the only way to determine that the operation is complete. Set an I/O timeout of at least 20 seconds.

Parameters

<name> String - Name of the power meter.

Examples

```
SYST:CONF:EDEV:PMAR:CAL "myDevice"  
system:configure:edevic:pmar:calibrate "myDevice"
```

Query Syntax Not Applicable

Default Not Applicable

SYSTEM:CONFigure:EDEVice:PMAR:FLIMit <name>,<value>

(Read-Write) Enable or disable the power meter min and max frequencies.

Parameters

<name> String - Name of the power meter.

<value> Boolean. State of min and max frequency. Choose from:

OFF or **0** - Min and max frequencies disabled.

ON or **1** - Min and max frequencies enabled.

Examples

```
SYST:CONF:EDEV:PMAR:FLIM "myDevice", 0  
system:configure:edevic:pmar:flimit "myDevice", ON
```

[See example program](#)

Query Syntax SYSTEM:CONFigure:EDEVice:PMAR:FLIMit? <name>

Return Type Boolean

Default OFF

SYSTem:CONFigure:EDEVice:PMAR:FMAXimum <name>,<value>

(Read-Write) Set and return the maximum frequency of the power meter.

Parameters

<name> String - Name of the power meter.

<value> Numeric - Max frequency in Hz.

Examples

```
SYST:CONF:EDEV:PMAR:FMAX "myDevice", 1e10
```

```
system:configure:edevic:pmar:fmaximum "myDevice", 3e9
```

[See example program](#)

Query Syntax SYSTem:CONFigure:EDEVice:PMAR:FMAXimum? <name>

Return Type Numeric

Default Not Applicable

SYSTem:CONFigure:EDEVice:PMAR:FMINimum <name>,<value>

(Read-Write) Set and return the minimum frequency of the power meter.

Parameters

<name> String - Name of the power meter.

<value> Numeric - Min frequency in Hz.

Examples

```
SYST:CONF:EDEV:PMAR:FMIN "myDevice", 1e10
```

```
system:configure:edevic:pmar:fminimum "myDevice", 3e9
```

[See example program](#)

Query Syntax SYSTem:CONFigure:EDEVice:PMAR:FMAXimum? <name>

Return Type Numeric

Default Not Applicable

SYSTem:CONFigure:EDEVice:PMAR:READing:COUNT <name>,<value>

(Read-Write) This command, along with SYST:CONF:EDEV:PMAR:READ:NTOL, allows for settling of the power sensor READINGS.

Set and return the maximum number of power readings that are taken at each stimulus point to allow for measurement settling. Each reading is averaged with the previous readings at that stimulus point.

When this average meets the Average:NTOLerance value or this number of readings has been made, the average is returned as the valid reading.

Parameters

<name> String - Name of the power meter.

<value> Number of readings. Choose a value between 1 and 25

Examples

```
SYST:CONF:EDEV:PMAR:READ:COUN "myDevice", 20  
system:configure:edevic:pmar:reading:count "myDevice", 10
```

[See example program](#)

Query Syntax SYSTem:CONFigure:EDEVice:PMAR:READing:COUNT? <name>

Return Type Numeric

Default 3

SYSTem:CONFigure:EDEVice:PMAR:READing:NTOLerance <name>,<value>

(Read-Write) This command, along with SYST:CONF:EDEV:PMAR:READ:COUN, allows for settling of the power sensor READINGS.

Each power reading is averaged with the previous readings at each stimulus point. When the average meets this nominal tolerance value, or the max number of readings has been made, the average is returned as the valid reading.

Parameters

<name> String - Name of the power meter.

<value> Power measurement settling tolerance value in dB. Choose any number between 0 and 5.

Examples

```
SYST:CONF:EDEV:PMAR:READ:NTOL "myDevice", .5  
system:configure:edevic:pmar:reading:ntolerance "myDevice",.01
```

[See example program](#)

Query Syntax SYSTem:CONFigure:EDEVice:PMAR:READing:NTOLerance? <name>

Return Type Numeric

Default .05

SYSTem:CONFigure:EDEVice:PMAR:SENSor <name>,<value>

(Read-Write) Sets and returns the power sensor channel (1 or 2) to be used. This performs the same function as the **Use this sensor only** checkbox.

Parameters

<name> String - Name of the power meter.

<value> Power Meter channel.

1 - Channel A

2 - Channel B

Examples

```
SYST:CONF:EDEV:PMAR:SENS "myDevice",2
```

```
system:configure:edev:pmar:sensor "myDevice",1
```

[See example program](#)

Query Syntax SYSTem:CONFigure:EDEVice:PMAR:SENSor? <name>

Return Type Numeric

Default 1

SYSTem:CONFigure:EDEVice:PMAR:TABLE:CFAC:DATA <name>,<value>[,value]

(Read-Write) Sets and returns the cal factor data for the power sensor.

Parameters

<name> String - Name of the power meter.

<value>[,value] Cal factor data in percent. For each frequency used with SYST:CONF:EDEV:PMAR:TABL:CFAC:FREQ, enter a cal factor number between 1 and 100.

Examples

```
SYST:CONF:EDEV:PMAR:TABL:CFAC:DATA "myDevice", 98,99,99
```

```
system:configure:edev:pmar:table:cfac:data "myDevice",  
97,97,97
```

[See example program](#)

Query Syntax SYSTem:CONFigure:EDEVice:PMAR:TABLE:CFAC:DATA? <name>

Return Type Numeric - one number per table segment.

Default Not Applicable

SYSTem:CONFigure:EDEVice:PMAR:TABLE:CFAC:FREQuency <name>,<value>[,value]

(Read-Write) Sets and returns the cal factor frequencies for the power sensor.

Parameters

<name> String - Name of the power meter.

<value>[,value] Cal factor frequencies in Hz.

Examples

```
SYST:CONF:EDEV:PMAR:TABLE:CFAC:FREQ "myDevice", 1e7,1e8,1e9
```

```
system:configure:edevic:pmar:table:cfac:frequency "myDevice",  
5e7,5e8,5e9
```

[See example program](#)

Query Syntax SYSTem:CONFigure:EDEVice:PMAR:TABLE:CFAC:FREQuency?<name>

Return Type Numeric - one number per table segment.

Default Not Applicable

SYSTem:CONFigure:EDEVice:PMAR:TABLE:LOSS:DATA <name>,<value>[,value]

(Read-Write) Sets and returns the power loss data for the power sensor.

Each table can contain up to 9999 segments. Values can also be loaded using the [Characterize Adapter macro](#).

Parameters

<name> String - Name of the power meter.

<value>[,value] Loss data in dB. POSITIVE values in dB are interpreted as LOSS. To compensate for gain, use negative values.

For each frequency used with SYST:CONF:EDEV:PMAR:TABLE:LOSS:FREQ, enter a cal factor number between 1 and 100.

Examples

```
SYST:CONF:EDEV:PMAR:TABLE:LOSS:DATA "myDevice",.01,.02,.03
```

```
system:configure:edevic:pmar:table:loss:data "myDevice",  
.04,.05,.06
```

[See example program](#)

Query Syntax SYSTem:CONFIgure:EDEVice:PMAR:TABLE:CFAC:DATA? <name>
Return Type Numeric - one number per table segment.
Default Not Applicable

SYSTem:CONFIgure:EDEVice:PMAR:TABLE:LOSS:FREQuency <name>,<value>[,value]

(Read-Write) Sets and returns frequencies for the power loss data.

Parameters

<name> String - Name of the power meter.
<value>[,value] Power Loss frequencies in Hz.

Examples

```
SYST:CONF:EDEV:PMAR:TABL:LOSS:FREQ "myDevice",1e7,1e8,1e9  
system:configure:edevic:pmar:table:loss:frequency  
"myDevice",5e7,5e8,5e9
```

[See example program](#)

Query Syntax SYSTem:CONFIgure:EDEVice:PMAR:TABLE:LOSS:FREQuency? <name>
Return Type Numeric - one number per table segment.
Default Not Applicable

SYSTem:CONFIgure:EDEVice:PMAR:TABLE:LOSS:STATe <name>,<value>

(Read-Write) Sets and returns whether to use the power loss table.

Parameters

<name> String - Name of the power meter.
<value> Boolean. State of the power loss table. Choose from:

OFF or **0** - Power loss table not used.

ON or **1** - Power loss table used.

Examples

```
SYST:CONF:EDEV:PMAR:TABL:LOSS:STAT "myDevice",1  
system:configure:edevic:pmar:table:loss:state "myDevice",1
```

[See example program](#)

Query Syntax SYSTem:CONFIgure:EDEVice:PMAR:TABLE:LOSS:STATe? <name>
Return Type Boolean

Default OFF

SYSTem:CONFigure:EDEVice:PMAR:TABLE:RFACtor <name>,<value>

(Read-Write) Sets and returns the reference cal factor for the power sensor.

Note: If the sensor connected to the power meter contains cal factors in EPROM (such as the Agilent E-series power sensors), those will be the cal factors used. The reference cal factor value associated with this command, and any cal factors entered into the PNA for that sensor channel, will not be used.

Parameters

<name> String - Name of the power meter.

<value> Reference cal factor in percent. Choose any number between 1 and 150.

Examples

```
SYST:CONF:EDEV:PMAR:TABL:RFAC "myDevice", 1  
system:configure:edevic:pmar:table:rfactor "myDevice", 1
```

[See example program](#)

Query Syntax SYSTem:CONFigure:EDEVice:PMAR:TABLE:RFACtor? <name>

Return Type Numeric

Default 100

SYSTem:CONFigure:EDEVice:PMAR:ZERO <name>[,SYNC,<value>]

(Write-only) Performs a zeroing of the PMAR device.

This command is always synchronous, so *OPC? is the only way to determine that the operation is complete. Set an I/O timeout of at least 20 seconds.

Agilent P-Series sensors do ONLY Internal zeroing. These, and Agilent USB power sensors when Internal is selected, do NOT require disconnecting from the measurement path before zeroing.

All other Agilent sensors do ONLY External zeroing.

Parameters

<name> String - Name of the power meter.

[,SYNC,<value>] Optional argument for use with power sensors that support both internal and external types of zeroing such as Agilent USB power sensors.

Choose from:

SYNC,INTernal - Internal zeroing. Power is automatically removed from the sensor input before zeroing occurs (Default setting).

SYNC,EXTernal - External zeroing. First remove the sensor input, then send this command. External zeroing is recommended for powers below -30 dBm with the U2000-Series sensors (-20 dBm for the H models).

Examples

```
SYST:CONF:EDEV:PMAR:ZERO "myDevice"
```

```
system:configure:edevic:pmar:zero "myDevice",sync,internal
```

Query Syntax Not Applicable

Default Not Applicable

Last Modified:

17-Aug-2012 Added Cal and Zero (9.80)

14-May-2012 Increased limit of loss table

SYSTem:CONF:EDEvice:PULSe Commands

Configures and makes settings for an external Agilent 81110A Pulse Generator.

SYST:CONF:EDEvice:PULSe

| [CHAN](#)
| [HAMP](#)
| [LAMP](#)
| [LIMP](#)
| [MMODE](#)
| [SIMP](#)

Click on a [blue](#) keyword to view the command details.

See Also

- Root [SYST:CONF:EDEV](#) commands
- All [Integrated Pulse App commands](#)
- Learn about: [Configure an External Pulse Generator](#)
- Learn about [Configure and External Device](#)
- [SYST:PREF:ITEM:EDEV:DPOL](#) - Determines whether External Devices remain activated or are de-activated when the PNA is Preset or when a Instrument State is recalled.
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

SYSTem:CONFigure:EDEvice:PULSe:CHAN <name>,<value>

(Read-Write) Sets and returns the output channel of the pulse generator.

Parameters

- <name> String - Name of the external pulse generator.
- <value> Pulse Generator output port. Choose from 1 or 2.

Examples

```
SYST:CONF:EDEV:PULS:CHAN "81110",1  
system:configure:edev:pulse:chan "myPG",2
```

Query Syntax SYSTem:CONFigure:EDEVice:DC:PULSe:CHAN? <name>

Return Type Numeric

Default 1

SYSTem:CONFigure:EDEVice:PULSe:HAMP <name>,<value>

(Read-Write) Sets and returns the High amplitude (voltage) of the pulse generator.

Parameters

- <name> String - Name of the external pulse generator.
- <value> Pulse Generator high amplitude voltage.

Examples

```
SYST:CONF:EDEV:PULS:HAMP "81110",3  
system:configure:edev:pulse:HAMP "myPG",4
```

Query Syntax SYSTem:CONFigure:EDEVice:DC:PULSe:HAMP? <name>

Return Type Numeric

Default 5

SYSTem:CONFigure:EDEVice:PULSe:LAMP <name>,<value>

(Read-Write) Pulse Generator low amplitude voltage.

Parameters

<name> String - Name of the external pulse generator.

<value> Pulse Generator low amplitude voltage.

Examples

```
SYST:CONF:EDEV:PULS:LAMP "81110",.2  
system:configure:edev:pulse:lamp "myPG",1
```

Query Syntax SYSTem:CONFigure:EDEVice:DC:PULSe:LAMP? <name>

Return Type Numeric

Default 0

SYSTem:CONFigure:EDEVice:PULSe:LIMP <name>,<value>

(Read-Write) Sets and returns the load impedance of the pulse generator.

Parameters

<name> String - Name of the external pulse generator.

<value> Pulse generator load impedance.

Examples

```
SYST:CONF:EDEV:PULS:LIMP "81110",52  
system:configure:edev:pulse:limp "myPG",49
```

Query Syntax SYSTem:CONFigure:EDEVice:DC:PULSe:LIMP? <name>

Return Type Numeric

Default 50

SYSTem:CONFigure:EDEVice:PULSe:MMODE <name>,<bool>

(Read-Write) Sets and returns the Master (On/Off) setting of the external pulse generator. The ON setting allows the external pulse generator to set the master clock frequency for the other pulse generators.

Parameters

<name> String - Name of the external pulse generator.

<bool> Master setting. Choose from:

ON or **1** - Use the external pulse generator becomes the master clock frequency.

OFF or **0** - Use the internal pulse generator as the master clock frequency.

Examples

```
SYST:CONF:EDEV:PULS:MMOD "81110",OFF
```

```
system:configure:edev:pulse:mmode "myPG",1
```

Query Syntax SYSTem:CONFIgure:EDEVice:DC:PULSe:MMODE? <name>

Return Type Boolean

Default OFF or 0

SYSTem:CONFIgure:EDEVice:PULSe:SIMP <name>,<value>

(Read-Write) Sets and returns the source impedance of the pulse generator.

Parameters

<name> String - Name of the external pulse generator.

<value> Pulse generator source impedance.

Examples

```
SYST:CONF:EDEV:PULS:SIMP "81110",52
```

```
system:configure:edev:pulse:simp "myPG",49
```

Query Syntax SYSTem:CONFIgure:EDEVice:DC:PULSe:SIMP? <name>

Return Type Numeric

Default 50

Last modified:

12-Mar-2014 New topic

SYSTEM:CORREction:INTerpolate:LINEar Commands

The five **SYSTEM:CORREction:INTerpolate:LINEar** commands are used as a sequence. They are not meant to be used independent of the others. The commands perform linear interpolation on a scalar data of (x,y) pairs based on a master set of x and y values to a new mapping based on a desired set of x values. The desired set of x values (or range) must fall within the master set of x values (or range) and can have a different number of points than the master data set.

Definition:

Linear interpolation operates by drawing a straight line between each two adjacent data points on the master (x,y) pairs that fall on either side of the new desired data point represented by (x',y'). in other words if

(xi, yi) represents data pairs on the master data set and (xj',yj') represents a data point on the interpolated result data set then:

$$X_i < X_{j'} < X_{i+1} \text{ and } Y_{j'} = Y_i + [(Y_{i+1} - Y_i)/(X_{i+1} - X_i)] (X_{j'} - X_i)$$

Note: The master data set must represent a function on the Cartesian coordinate system. In other words, for each x value in the master data set, there can be only one corresponding Y value.

There are five steps in the sequence:

1. SYSTEM:CORREction:INTerpolate:LINEar:INPut:X - loads in the master X values
2. SYSTEM:CORREction:INTerpolate: LINEar:INPut:Y - loads in the master Y values
3. SYSTEM:CORREction:INTerpolate: LINEar:OUTput:X - loads in the desired interpolated X values
4. SYSTEM:CORREction:INTerpolate: LINEar:CALCulate - calculates the interpolated Y values
5. SYSTEM:CORREction:INTerpolate: LINEar:OUTput:Y? - reads back the interpolated Y values.

Example

The following function uses the SYSTEM:CORREction:INTerpolate: LINEar commands:

```
Function InterpolateData_Single(inputX() As Double, inputY() As Single, outputX()  
As Double, ByRef interpData() As Single)  
  
    x = visa_io.ag_send_binBlock64("SYST:CORR:INT:LIN:INP:X ", inputX)  
  
    x = visa_io.ag_send_binBlock("SYST:CORR:INT:LIN:INP:Y ", inputY)  
  
    x = visa_io.ag_send_binBlock64("SYST:CORR:INT:LIN:OUTP:X ", outputX)
```

```

x = visa_io.ag_send_rd("*OPC?")

x = visa_io.ag_send_wait("SYST:CORR:INT:LIN:CALC")

x = visa_io.ag_send_rd("*OPC?")

interpData = visa_io.ag_send_rd_binBlock("SYST:CORR:INT:LIN:OUTP:Y?")

CheckError

End Function

'Here is a code snippet that uses the function above.
'copy the B-Response Error term from one calset to another.
'The source calset has a super set stimulus and
'and the receiving calset has a subset stimulus

Dim BResp_freqList() As Double

Dim BResp_Re() As Single

Dim BResp_Im() As Single

GetErrorTerm_noChan BResp_Calset, "ResponseTracking(B)", BResp_freqList,
BResp_Re, BResp_Im

Dim Noise_freqList() As Double

GetCalsetStimulus calsetName, Noise_freqList, 1, "Noise Figure Cold Source"
' Response Stimulus Range

Dim BResp_Re_interp() As Single

Dim BResp_Im_interp() As Single

InterpolateData_Single BResp_freqList, BResp_Re, Noise_freqList,
BResp_Re_interp

InterpolateData_Single BResp_freqList, BResp_Im, Noise_freqList,

```

BResp_Im_interp

```
PutErrorTerm channel, calsetName, "ResponseTracking(B)", BResp_Re_interp,  
BResp_Im_interp
```

Last Modified:

28-Jan-2011 MX New topic

System:FIFO Commands

The 4 GB FIFO data buffer is available with Option 118 on the [PNA-X](#) and [N5264A](#). These commands control data in and out of FIFO data buffer. The FIFO can be emptied as it is being filled, which means that the PNA can be used to acquire an infinite amount of data.

The data placed into the FIFO is the raw data after averaging and ratioing has been applied, but prior to any calibration, formatting, or data analysis functions.

SYSTem:FIFO

DATA

| **CLEar**

| **COUNT?**

[:STATe]

Click on a [blue](#) keyword to view the command details.

See Also

- [FIFO and other Antenna Features](#)
 - [Fast CW command](#)
 - [FIFO Example Program](#)
 - [Synchronizing the PNA and Controller](#)
 - [SCPI Command Tree](#)
-

SYSTem:FIFO:DATA? <dpoints>

(Read-only) Reads the next specified number of data points from the FIFO buffer. Each data point is returned as a real/imaginary pair. Data is cleared as it is read.

Parameters

<dPoints> Number of data points to read. An error is returned if the amount of requested data is larger than the available data.

Examples

```
SYST:FIFO:DATA? 1e6  
system:fifo:data? 1e3
```

Return Type Use [FORMat:DATA](#) to change the data type (<REAL,32>, <REAL,64> or <ASCIi,0>). For best results, use REAL,32

Use [FORMat:BORDER](#) to change the byte order. Use "NORMAL" when transferring a binary block from LabView or Vee. For other programming languages, you may need to "SWAP" the byte order.

Each data point is returned as a real/imaginary pair.

Default Not applicable

SYSTem:FIFO:DATA:CLEar

(Write-only) Clears the data from the FIFO buffer.

Parameters None

Examples

```
SYST:FIFO:DATA:CLE  
system:fifo:data:clear
```

Return Type None

Default Not applicable

SYSTem:FIFO:DATA:COUNt?

(Read-only) Returns the total number of data points in the FIFO buffer.

Parameters None

Examples

```
SYST:FIFO:DATA:COUN?  
' returns 5.07e6
```

Return Type Numeric

Default Not applicable

SYSTem:FIFO[:STATE] <bool>

(Write-Read) Sets and returns the state of data storage to the FIFO buffer. Syst:Preset or an instrument state recall also ends storage to the FIFO buffer. The FIFO buffer is cleared when set to OFF.

Parameters

<bool> FIFO buffer state. Choose from:

ON or 1 Data is stored in the FIFO buffer.

OFF or 0 Data is NOT stored in the FIFO buffer.

Examples

```
SYST:FIFO 1  
system:fifo:state off
```

Query Syntax SYSTem:FIFO[:STATE]?

Return Type Boolean

Default 0 OFF

Last Modified:

6-Sep-2012 Added notes to syst:fifo:data? (JE)

Fixed tree typo

6-Mar-2009 MX New topic

System Preferences Commands

Sets and reads the PNA Preferences settings.

```
SYSTem:PREFerences
| DEFault
| ITEM
| EEXTrapolate
| EDEV: DPOLicy
| GDELay:TWOPoint
| MRU
| OFFSet
| RCV
| SRC
| PRESet:POWer:STATe
| PSRTrace
| RECeivers
| CERRor
| OVERload:POWer
| RETRace:POWer
| RTOF
| SWITCh:DEF
```

Click on a [blue](#) keyword to view the command details.

See Also

- [SENS:CORRection:PREFerences](#)
- [Learn about PNA Preferences](#)
- [Example Programs](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

SYSTem:PREFerences:DEFault

(Write-only) Resets the PNA preferences to their default settings. Some default settings vary depending on the PNA Model. [Learn more about PNA Preferences.](#)

Examples `SYST:PREF:DEF`
`system:preferences:default`

Query Syntax Not Applicable

Default Not Applicable

SYSTEM:PREferences:ITEM:EDEV:DPOLicy <bool>

(Read-Write) Set and return whether External Devices remain activated or are de-activated when the PNA is Preset or when a Instrument State is recalled.

This setting remains until changed again using this command, or until the hard drive is changed or reformatted.

Parameters

<bool> Choose from:

OFF (0) External devices **remain active** when the PNA is Preset or when a Instrument State is recalled.

ON (1) External devices are **de-activated** ([SYST:CONF:EDEV:STAT](#) to OFF) when the PNA is Preset or when a Instrument State is recalled.

Examples `SYST:PREF:ITEM:EDEV:DPOL 1`
`system:preferences:item:edev:dpolicy OFF`

Query Syntax SYSTEM:PREferences:ITEM:EDEV:DPOLicy?

Return Type Boolean

Default ON or 1

SYSTEM:PREferences:ITEM:EEXTrapolate <bool>

(Read-Write) Sets whether a Swept IMD or IMDx calibration can exceed the stop frequency limit of an ECal module. [Learn more.](#)

Parameters

<bool> Choose from:

ON (1) Allow extrapolation.

OFF (0) Do NOT allow extrapolation.

Examples

```
SYST:PREF:ITEM:EEXT 1
```

```
system:preferences:item:eextrapolate OFF
```

Query Syntax SYSTem:PREFErences:ITEM:EEXTrapolate?

Return Type Boolean

Default OFF

SYSTem:PREFErences:ITEM:GDELaY:TWOPoint <bool>

(Read-Write) Sets the default group delay aperture setting. [Learn more about group delay aperture.](#)

Parameters

<bool> Choose from:

ON (1) Sets default group delay aperture to 2 points.

OFF (0) Sets default group delay aperture to 11 points.

Examples

```
SYST:PREF:ITEM:GDELaY:TWOPoint 1
```

```
system:preferences:item:gdelay:twopoint OFF
```

Query Syntax SYSTem:PREFErences:ITEM:GDELaY:TWOPoint?

Return Type Boolean

Default OFF

SYSTem:PREFErences:ITEM:MRU <bool>

(Read-Write) Set and return whether to list files for recall on softkeys by most-recently used or alphabetically.

Parameters

<bool> Choose from:

ON (1) – Recall softkeys show most recently-used files.

OFF (0) – Recall softkeys show alphabetically-ordered files.

Examples

```
SYST:pref:ITEM:MRU 1
```

```
system:preferences:item:mru OFF
```

Query Syntax SYSTem:PREFerences:ITEM:MRU?

Return Type Boolean

Default OFF (0)

SYSTem:PREFerences:ITEM:OFFSet:RCV <bool>

(Read-Write) Set and return whether to offset the test port receivers by the amount of receiver attenuation. [Learn more.](#)

To send this command using the PNA front panel, open the [GPIB Command Processor Console](#), then type either of the following examples at the command prompt. Then type the Query Syntax and press enter to be sure the PNA took the command.

This setting remains until changed again using this command, or until the hard drive is changed or reformatted.

Parameters

<bool> Choose from:

ON (1) Offset the test port receivers

OFF (0) Do NOT offset the test port receivers

Examples

```
SYST:pref:ITEM:OFFS:RCV 1
```

```
system:preferences:item:offset:rcv OFF
```

Query Syntax SYSTem:PREFerences:ITEM:OFFSet:RCV?

Return Type Boolean

Default PNA-L and E836xB: **OFF** (does NOT offset the display).

PNA-X: **ON** (offsets the display).

SYSTem:PREFerences:ITEM:OFFSet:SRC <bool>

(Read-Write) Set and return whether to offset the reference receiver by the amount of source attenuation. [Learn more.](#)

To send this command using the PNA front panel, open the [GPIB Command Processor Console](#), then type either of the following examples at the command prompt. Then type the Query Syntax and press enter to be sure the PNA took the command.

This setting remains until changed again using this command, or until the hard drive is changed or reformatted.

Parameters

<bool> Choose from:

ON (1) Offset the reference receivers.

OFF (0) Do NOT Offset the reference receivers.

Examples

```
SYST:PREF:ITEM:OFFS:SRC 1
system:preferences:item:offset:src OFF
```

Query Syntax SYSTem:PREFerences:ITEM:OFFSet:SRC?

Return Type Boolean

Default All models: **ON** (offset the display).

SYSTem:PREFerences:ITEM:PRESet:POWer[:STATe] <char>

(Read-Write) Set and return the Preset Power ON/OFF state. [Learn more.](#)

This setting remains until changed again using this command, or until the hard drive is changed or reformatted.

Parameters

<char> Choose from:

ON - Instrument Preset always turns RF power ON.

AUTO - When the current power setting is OFF, leave power OFF after Preset. When the current power setting is ON, turn power ON after Preset.

Examples

```
SYST:PREF:ITEM:PRE:POW ON
system:preferences:item:preset:power:state auto
```

Query Syntax SYSTem:PREFerences:ITEM:PRESet:POWer[:STATe]?

Return Type Character

Default ON

SYSTEM:PREferences:ITEM:PSRTrace <char>

(Read-Write) At the end of a power sweep, while waiting to trigger the next sweep, maintain source power at either the start power level or at the stop power level.

To send this command using the PNA front panel, open the [GPIB Command Processor Console](#), then type either of the following examples at the command prompt. Then type the Query Syntax and press enter to be sure the PNA took the command.

This setting remains until changed again using this command, or until the hard drive is changed or reformatted.

Parameters

<char> Choose from:

START - Maintain source power at the start power level.

STOP - Maintain source power at the stop power level.

Examples

```
SYST:PREF:ITEM:PSRT STOP  
system:preferences:item:psrtrace start
```

Query Syntax SYSTEM:PREferences:ITEM:PSRTrace?

Return Type Character

Default START

SYSTEM:PREferences:ITEM:RECEivers:CERRor[:STATE] <bool>

(Read-Write) Set and return whether to display receiver overload warnings. [Learn more.](#)

Parameters

<bool> Choose from:

ON (1) Display overload warnings,

OFF (0) Do NOT display overload warnings.

Examples

```
SYST:PREF:ITEM:REC:CERR 1
```

```
system:preferences:item:receivers:error:state OFF
```

Query Syntax SYSTem:PREFerences:ITEM:RECEivers:CERRor[:STATe]?

Return Type Boolean

Default ON

SYSTem:PREFerences:ITEM:RECEivers:OVERload:POWER[:STATe] <bool>

(Read-Write) Set and return whether to turn source power OFF when a receiver is overloaded. [Learn more.](#)

Parameters

<bool> Choose from:

ON (1) Turn OFF source power to ALL ports when a receiver is overloaded.

OFF (0) Power remains ON when a receiver is overloaded.

Examples

```
SYST:PREF:ITEM:REC:OVER:POW 1
```

```
system:preferences:item:receivers:overload:power:state OFF
```

Query Syntax SYSTem:PREFerences:ITEM:RECEivers:OVERload:POWER[:STATe]?

Return Type Boolean

Default OFF (0)

SYSTem:PREFerences:ITEM:RETRace:POWER <char>

(Read-Write) For single-band frequency or segment sweeps ONLY, specify whether to turn RF power ON or OFF during a retrace. [Learn more about RF power during sweep retrace.](#)

To send this command using the PNA front panel, open the [GPIB Command Processor Console](#), then type either of the following examples at the command prompt. Then type the Query Syntax and press enter to be sure the PNA took the command.

This setting remains until changed using this command, or until the hard drive is changed or reformatted.

Parameters

<char> Choose from:

AUTO: Power is left ON during retrace of single-band frequency or segment sweeps ONLY.

OFF: Power is turned OFF during retrace of single-band frequency or segment sweeps ONLY.

Examples

```
SYST:PREF:ITEM:RETR:POW OFF
system:preferences:item:retrace:power auto
```

Query Syntax SYSTem:PREFErences:ITEM:RETRace:POWER?

Return Type Character

Default AUTO

SYSTem:PREFErences:ITEM:RTOF <bool>

(Read-Write) Set and return whether to display limit line failures as red trace segments or red data points (dots). [Learn more.](#)

Parameters

<bool> Choose from:

ON (1) Display failures as red trace segments. (Red Trace On Fail).

OFF (0) Display failures as red data points (dots).

Examples

```
SYST:PREF:ITEM:RTOF 1
system:preferences:item:rtof OFF
```

Query Syntax SYSTem:PREFErences:ITEM:RTOF?

Return Type Boolean

Default OFF

SYSTem:PREFerences:ITEM:SWITch:DEF <string>, <int>

(Read-Write) Sets the default setting for the Noise Tuner switch. This is the setting that occurs when a new channel is created. [Learn more.](#)

This command will return an error on PNA models with a built-in Noise tuner.

To send this command using the PNA front panel, open the [GPIB Command Processor Console](#), then type either of the following examples at the command prompt. Then type the Query Syntax and press enter to be sure the PNA took the command.

This setting remains until changed using this command, or until the hard drive is changed or reformatted.

Parameters

<string> Name of the switch to set. Choose from:

- "Port1NoiseTuner"

<int> Value to set. Choose from:

0 Sets the default (preset) to INTERNAL

1 Sets the default (preset) to EXTERNAL

Examples

```
SYST:PREF:ITEM:SWIT:DEF "Port1NoiseTuner" 1 'Write
system:preferences:item:switch:def? "Port1NoiseTuner" 'Read
```

Query Syntax SYSTem:PREFerences:ITEM:SWITch:DEF? <switch>

Return Type Integer

Default **1** (External)

Last Modified:

1-Aug-2011	Added MRU
30-Aug-2010	Added receiver overload
30-Aug-2010	New topic started from System level

Trigger Commands

Controls External Triggering on PNA-X and N522x models.

TRIGger:

AUX

| COUNT

CHANnel:AUX

| DELay

| DURation

| [ENABle]

| HANDshake

| INTerval

| IPOLarity

| OPOLarity

| POSition

| TYPE

DELay

PREFerence

| AIGLobal

READy:POLarity

[SEQuence]

| LEVel

| ROUTE

| INPut

| READy

| SCOPe

| SOURce

| SLOPe

| **TYPE**

Click on a [blue](#) keyword to view the command details.

Red commands are superseded.

See Also

- Example program [Triggering the PNA](#)
 - [See other SCPI Triggering commands](#)
 - [Learn about External / Aux Triggering](#)
 - [Synchronizing the PNA and Controller](#)
 - [SCPI Command Tree](#)
-

TRIGger:AUXiliary:COUNT?

(Read-only) Returns the number of AUX trigger input / output connector pairs in the instrument.

Parameters

Examples

```
TRIG:AUX:COUN?  
trigger:auxiliary:count?
```

Return Type Numeric

Default Not Applicable

TRIGger:CHANnel<ch>:AUXiliary<n>:DELay <num>

(Read-Write) Specifies the delay that should be applied by the PNA after the Aux trigger input is received and before the acquisition is made.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<n> Rear panel connectors used to send or receive signals.

- PNA-X - choose from **1** ([AUX TRIG 1 IN](#)) or **2** ([AUX TRIG 2 IN](#))
- All other models: choose **1**.

If unspecified, value is set to 1.

<num> Delay value in seconds. Choose a value between 0 and 3.0 seconds.

Examples

```
TRIG:CHAN:AUX:DEL .5  
trigger:channel2:aux2:delay 1.5
```

Query Syntax TRIGger:CHANnel<ch>:AUXiliary<n>:DELay?

Return Type Numeric

Default 0

TRIGger:CHANnel<ch>:AUXiliary<n>:DURation <num>

(Read-Write) Specifies the width of the output pulse, which is the time that the Aux trigger output will be asserted.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<n> Rear panel connector used to send or receive signals.

Choose from **1** ([AUX TRIG 1](#)) or **2** ([AUX TRIG 2](#))

If unspecified, value is set to 1.

<num> Duration value in seconds. Choose a value between 1us (1E-6) and 1

Examples

```
TRIG:CHAN:AUX:DUR .1  
trigger:channel2:aux2:duration .01
```

Query Syntax TRIGger:CHANnel<ch>:AUXiliary<n>:DURation?

Return Type Numeric

Default 1E-6

TRIGger:CHANnel<ch>:AUXiliary<n>[:ENABLE] <bool>

(Read-Write) Turns ON / OFF the trigger output.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<n> Rear panel connector used to send or receive signals.

Choose from **1** ([AUX TRIG 1](#)) or **2** ([AUX TRIG 2](#))

If unspecified, value is set to 1.

<bool> **ON** (or 1) - turns trigger output ON.

OFF (or 0) - turns trigger output OFF.

Examples

```
TRIG:CHAN:AUX 1
```

```
trigger:channel2:aux2:enable off
```

Query Syntax TRIGger:CHANnel<ch>:AUXiliary<n>[:ENABLE]?

Return Type Boolean

Default OFF

TRIGger:CHANnel<ch>:AUXiliary<n>:HANDshake <bool>

(Read-Write) Turns handshake ON / OFF.

To enable handshake, the main trigger enable must also be set using [TRIG:CHAN:AUX:ENAB](#).

When ON, PNA waits indefinitely for the input line to be asserted before continuing with the acquisition. When OFF, the PNA acquires data without waiting.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<n> Rear panel connector used to send or receive signals.

Choose from **1** ([AUX TRIG 1](#)) or **2** ([AUX TRIG 2](#))

If unspecified, value is set to 1.

<bool> **ON** (or 1) - turns handshaking ON.

OFF (or 0) - turns handshaking OFF.

Examples TRIG:CHAN:AUX:HAND 1

```
trigger:channel2:aux2:handshake off
```

Query Syntax TRIGger:CHANnel<ch>:AUXiliary<n>:HANDshake?

Return Type Boolean

Default OFF

TRIGger:CHANnel<ch>:AUXiliary<n>:INTerval <char>

(Read-Write) Specifies how often a trigger output signal is sent.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<n> Rear panel connector used to send or receive signals.

Choose from 1 ([AUX TRIG 1](#)) or 2 ([AUX TRIG 2](#))

If unspecified, value is set to 1.

<char> Choose from:

- **POINT** Trigger signal is sent every data point. (effectively the same as [Point sweep](#))
- **SWEep** Trigger signal is sent once every sweep.

Examples TRIG:CHAN:AUX:INT POI

```
trigger:channel2:aux2:interval sweep
```

Query Syntax TRIGger:CHANnel<ch>:AUXiliary<n>:INTerval?

Return Type Character

Default SWEep

TRIGger:CHANnel<ch>:AUXiliary<n>:IPOLarity <char>

(Read-Write) Specifies the polarity of the trigger IN signal to which the PNA will respond.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<n> Rear panel connector used to send or receive signals.

Choose from **1** ([AUX TRIG 1](#)) or **2** ([AUX TRIG 2](#))

If unspecified, value is set to 1.

<char> Choose from:

- **POSitive** PNA responds to leading edge or HIGH level
- **NEGative** PNA responds to trailing edge or LOW level.

Set Edge or Level triggering using [TRIG:CHAN:AUX:TYPE](#)

Examples

```
TRIG:CHAN:AUX:IPOL POS
```

```
trigger:channel2:aux2:ipolarity negative
```

Query Syntax TRIGger:CHANnel<ch>:AUXiliary<n>:IPOLarity?

Return Type Character

Default NEGative

TRIGger:CHANnel<ch>:AUXiliary<n>:OPOLarity <char>

(Read-Write) Specifies the polarity of the Aux Output signal being supplied by the PNA.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<n> Rear panel connector used to send or receive signals.

Choose from **1** ([AUX TRIG 1](#)) or **2** ([AUX TRIG 2](#))

If unspecified, value is set to 1.

<char> Choose from:

- **POSitive** PNA sends positive going pulse.
- **NEGative** PNA sends negative going pulse.

Examples

```
TRIG:CHAN:AUX:OPOL NEG
trigger:channel2:aux2:opolarity positive
```

Query Syntax TRIGger:CHANnel<ch>:AUXiliary<n>:OPOLarity?**Return Type** Character**Default** NEGative**TRIGger:CHANnel<ch>:AUXiliary<n>:POSition <char>****(Read-Write)** Specifies whether the aux trigger out signal is sent BEFore or AFTer the acquisition.**Parameters**

<ch> Any existing channel number. If unspecified, value is set to 1.

<n> Rear panel connector used to send or receive signals.

Choose from **1** ([AUX TRIG 1](#)) or **2** ([AUX TRIG 2](#))

If unspecified, value is set to 1.

<char> Choose from:

- **BEFore** Use if the external device needs to be triggered before the data is acquired, such as a power meter.
- **AFTer** Use if the external device needs to be triggered just after data has been acquired, such as an external source. This could be more efficient since it allows the external device to get ready for the next acquisition at the same time as the PNA.

Examples

```
TRIG:CHAN:AUX:POS BEF
trigger:channel2:aux2:position after
```

Query Syntax TRIGger:CHANnel<ch>:AUXiliary<n>:POSition?**Return Type** Character**Default** AFTer**TRIGger:CHANnel<ch>:AUXiliary<n>:TYPE <char>**

(Read-Write) Specifies the type of Aux input detection that the PNA will employ.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<n> Rear panel connector used to send or receive signals.

Choose from **1** ([AUX TRIG 1](#)) or **2** ([AUX TRIG 2](#))

If unspecified, value is set to 1.

<char> Choose from:

EDGE PNA responds to the leading edge of a signal

LEVel PNA responds to the level (HIGH or LOW) of a signal

Examples

```
TRIG:CHAN:AUX:TYPE EDGE
```

```
trigger:channel2:aux2:type level
```

Query Syntax TRIGger:CHANnel<ch>:AUXiliary<n>:TYPE?

Return Type Character

Default EDGE

TRIGger:DELay <num>

(Read-Write) Sets and reads the trigger delay for ALL channels (globally). This delay is only applied while [TRIG:SOURce](#) = EXTernal and [TRIG:SCOP](#) = ALL After an external trigger is applied, the start of the sweep is held off for an amount of time equal to the delay setting plus any inherent latency.

To apply a trigger delay for the specified channel ONLY, use [SENS:SWE:TRIG:DELay](#)

Parameters

<num> Delay value in seconds. Choose from 0 to 107.

Examples

```
TRIG:DEL .0003
```

```
Sets the trigger delay to 300 microseconds. The sweep will not start until approximately 300 microseconds after an external trigger is applied.
```

Query Syntax TRIGger:DELay?

Return Type Numeric

Default 0

TRIGger:PREFerence:AIGLobal <bool>

(Read-Write) Sets the Trigger OUT behavior to either Global or Channel. [Learn more about this setting.](#)

This command will cause the PNA to Preset.

This setting remains until changed again using this command, or until the hard drive is changed or reformatted.

To send this command using the PNA front panel, open the [GPIB Command Processor Console](#), then type either of the following examples at the command prompt. Then type the Query Syntax and press enter to be sure the PNA took the command.

Parameters

<bool> Choose from:

- **ON** (or 1) - Trigger properties apply to ALL channels (Global).
 - Allows use of [CONT:SIGNal](#) command to configure the external trigger properties.
 - "Per Point" trigger property is not settable. Use the channel's [Point trigger](#) setting.
- **OFF** (or 0) - External Trigger properties apply to each channel independently.
 - Must use [TRIG:CHAN:AUX](#) commands to configure the external trigger properties. [CONT:SIGNal](#) will NOT work.
 - "Per Point" trigger output property is set using the channel's [Point trigger](#) setting **AND** [TRIG:CHAN:AUX:INTerval](#).

Examples

```
TRIG:PREF:AIGL 1  
trigger:preference:aiglobal 0
```

Query Syntax TRIGger:PREFerence:AIGLobal?

Return Type Boolean

Default 0

TRIGger:READy:POLarity <char>

(Read-Write) Specifies the polarity of Ready for Trigger output.

All existing Ready for Trigger outputs are configured simultaneously with this command.

Parameters

<char> **LOW** - Outputs a TTL low when the PNA is ready for trigger.

HIGH - Outputs a TTL high when the PNA is ready for trigger.

Examples

```
TRIG:READ:POL HIGH
trigger:ready:polarity low
```

Query Syntax TRIGger:READy:POLarity?

Return Type Character

Default Low

TRIGger[:SEquence]:LEVel <char> - Superseded

This command is replaced with [CONTRol:SIGNal](#)

(Read-Write) Triggers either on a **High or Low** level trigger signal. This setting only has an effect when [TRIG:SOURce EXTernal](#) is selected.

Parameters

<char> Choose from:

- **HIGH** - analyzer triggers on TTL **High**
- **LOW** - analyzer triggers on TTL **Low**

Examples

```
TRIG:LEV HIGH
trigger:sequence:level low
```

Query Syntax TRIGger[:SEquence]:LEVel?

Return Type Character

Default LOW

TRIGger[:SEquence]:ROUTE:INPut <char>

(Read-Write) Specifies the connector to use for the external trigger input.

Parameters

<char> Choose from:

MAIN - Meas Trig In BNC

MATH - material handler I/O Pin 18

Examples

```
TRIG:ROUTE:INP MAIN
```

```
trigger:sequence:route:input main
```

Query Syntax TRIGger[:SEquence]:ROUTE:INPut?

Return Type Character

Default MAIN

TRIGger[:SEquence]:ROUTE:READY <char>

(Read-Write) Specifies the connector to use for the trigger OUT ready line.

Parameters

<char> Choose from:

MAIN - [Meas trig ready](#)

MATH - [Material handler pin 21](#)

Examples

```
TRIG:ROUTE:READ main
```

```
trigger:sequence:route:ready math
```

Query Syntax TRIGger[:SEquence]:ROUTE:READY?

Return Type Character

Default MAIN

TRIGger[:SEquence]:SCOPE <char>

(Read-Write) Specifies whether a trigger signal is sent to all channels or only the current channel.

See [Triggering the PNA using SCPI](#).

Parameters

<char> Choose from:

- **ALL** - trigger signal is sent to all channels. Also sets [SENS:SWEep:TRIG:POINT OFF](#) on **ALL** channels.
- **CURRent** - trigger signal is sent to only one channel at a time. With each trigger signal, the channel is incremented to the next triggerable channel.

Examples

```
TRIG:SCOP ALL
trigger:sequence:slope current
```

Query Syntax TRIGger[:SEquence]:SCOPE?

Return Type Character

Default ALL

TRIGger[:SEquence]:SLOPe <char>

(Read-Write) Specifies the polarity expected by the external trigger input circuitry. Also specify [TRIG:TYPE](#) (Level|Edge).

See [Triggering the PNA using SCPI](#).

Parameters

<char> Choose from:

- **POSitive** (rising Edge) or High Level
- **NEGative** (falling Edge) or Low Level

Examples

```
TRIG:SLOP NEG
trigger:sequence:slope positive
```

Query Syntax TRIGger[:SEquence]:SLOPe?

Return Type Character

Default POSitive

TRIGger[:SEquence]:SOURce <char>

(Read-Write) Sets the source of the sweep trigger signal. This command is a super-set of [INITiate:CONTinuous](#) which can NOT set the source to External.

See [Triggering the PNA using SCPI](#).

Parameters

<char> Choose from:

- **EXTernal** - external (rear panel) source.
- **IMMediate** - internal source sends continuous trigger signals
- **MANual** - sends one trigger signal when manually triggered from the front panel or [INIT:IMM](#) is sent.

Examples

```
TRIG:SOUR EXT  
trigger:sequence:source immediate
```

Query Syntax TRIGger[:SEquence]:SOURce?

Return Type Character

Default IMMediate

TRIGger[:SEquence]:TYPE <char>

(Read-Write) Specifies the type of EXTERNAL trigger input detection used to listen for signals on the Meas Trig IN connectors. Edge triggers are most commonly used.

Parameters

<char> Choose from:

EDGE PNA responds to the rising and falling edge of a signal.

LEVel PNA responds to a level (HIGH or LOW).

Use [TRIG:SLOPe](#) to specify Rising or falling - High or Low.

Examples

```
TRIG:TYPE EDGE  
trigger:sequence:type level
```

Query Syntax TRIGger[:SEquence]:TYPE?

Return Type Character

Default LEVel

Last modified:

12-Mar-2014 Fixed trig:type
23-Jan-2013 Removed <ch> argument from several commands
12-Apr-2012 Removed reference to old models-big syntax fix
28-Sep-2009 Fixed CHAN:AUXiliary commands
11-Feb-2009 Added TRIG:SLOPe
14-Mar-2008 Added READy:POL command
22-Feb-2008 Clarified AIGL command
24-Apr-2007 Clarified trigger source and scope
15-Feb-2007 MX Updated for AUX triggering

Catalog Measurements using SCPI

This Visual Basic Program does the following:

- Catalogs the currently defined measurements, windows, and traces
- Selects a measurement for further definition
- Adds a Title to the window

To run this program, you need:

- An established [GPIB interface connection](#)

[See Other SCPI Example Programs](#)

```
Dim Meas as String
Dim Win as String
Dim Trace as String

'Read the current measurements in Channel 1
GPIB.Write "CALCulate1:PARAMeter:CATalog?"
Meas = GPIB.Read
MsgBox ("Ch1 Measurments: " & Meas)

'Read the current windows
GPIB.Write "DISPlay:CATalog?"
Win = GPIB.Read
MsgBox ("Windows: " & Win)

'Read current traces in window 1
GPIB.Write "DISPlay:WINDow1:CATalog?"
Trace = GPIB.Read
MsgBox ("Traces in Window1: " & Win)
```

Channels, Windows, and Measurements using SCPI

SOURCE and most **SENSE** commands act on the **channel** that is specified in the command. Channel 1 is default if not specified.

Most **DISPLAY** commands act on the **window and trace** specified in the command. Window1 and Trace1 are default if not specified.

CALCulate commands act on the **selected measurement** in the specified channel. Select the measurement for each channel using `CALCulate<channel number>:PARAMeter:SElect <meas name>`. You can select one measurement in each channel.

The following Visual Basic program does the following:

- Presets the analyzer
- Create 2 windows
- Create 2 Measurements
- Feed the measurements to windows / traces
- Change frequency ranges for channels
- Select both measurements
- Turn marker 1 ON for each measurement

To run this program, you need:

- An established [GPIB interface connection](#)

[See Other SCPI Example Programs](#)

```
GPIB.Write "SYSTem:PRESet"  
  
'Create Measurements  
GPIB.Write "CALCulate1:PARAMeter:DEFine:EXT 'Meas1',S11"  
GPIB.Write "CALCulate2:PARAMeter:DEFine:EXT 'Meas2',S21"  
  
' Turn on windows - creates if new  
GPIB.Write "DISPlay:WINDow1:STATE ON"  
GPIB.Write "DISPlay:WINDow2:STATE ON"  
  
'Associate ("FEED") the measurement name('Meas1') to WINDow(1), and give the new  
TRACe a number(1).
```

```
GPIB.Write "DISPlay:WINDow1:TRACe1:FEED 'Meas1'"
GPIB.Write "DISPlay:WINDow2:TRACe2:FEED 'Meas2'"

'Change each channel's frequency range
GPIB.Write "SENSe1:FREQuency:SPAN 1e9"
GPIB.Write "SENSe2:FREQuency:SPAN 2e9"

'Select both measurements
GPIB.Write "CALCulat1:PARAmeter:SELEct 'Meas1'"
GPIB.Write "CALCulate2:PARAmeter:SELEct 'Meas2'"

'Turn marker 1 ON for each measurement
GPIB.Write "CALCulat1:MARKer:STATE ON"
GPIB.Write "CALCulate2:MARKer:STATE ON"
```

PNA as Controller and Talker / Listener

This Visual Basic Program uses VISA to do the following:

- Control the PNA using a VISA LAN Client interface on the PNA.
- Control another instrument using the PNA as GPIB controller.
- Queries both the analyzer and other instrument to identify themselves with *IDN?

Note: This program can be modified to work from a remote PC to control both instruments. In that case, set up the PNA to be a talker/listener.

To run this program, you need to do the following:

- Add module **visa32.bas** to the VB project.
- [Configure the PNA for VISA / SICL](#)
- Set up the PNA to be GPIB system controller.
 1. On the **System** menu, point to **Configure**. Click **SICL / GPIB**
 2. Click **System Controller**
- Connect another instrument to the analyzer through a GPIB cable with Primary address of 13 on GPIB0 interface

[See Other SCPI Example Programs](#)

```
Sub main()  
  
'This application run from onboard the PNA  
'can control both the PNA and another GPIB instrument.  
'  
'To run this program the module visa32.bas must be added  
'to the project.  
  
'VISA function status return code  
Dim status As Long  
'Session to Default Resource Manager  
Dim defRM As Long  
'Session to instrument  
Dim viPNA As Long  
'Session to other GPIB instrument  
Dim viInstrument As Long  
'String to hold results  
Dim strRes As String * 200  
On Error GoTo ErrorHandler
```

```

status = viOpenDefaultRM(defRM)
If (status < VI_SUCCESS) Then GoTo VisaErrorHandler

'Open the session to the PNA
status = viOpen(defRM, "GPIB1::16::INSTR", 0, 0, viPNA)
If (status < VI_SUCCESS) Then GoTo VisaErrorHandler

'Ask for the PNA's ID.
status = viVPrintf(viPNA, "*IDN?" + Chr$(10), 0)
If (status < VI_SUCCESS) Then GoTo VisaErrorHandler

'Read the ID as a string.
status = viVScanf(viPNA, "%t", strRes)
If (status < VI_SUCCESS) Then GoTo VisaErrorHandler
'Display the results
MsgBox "PNA is: " + strRes

'Open the session to the other instrument
status = viOpen(defRM, "GPIB0::13::INSTR", 0, 0, viInstrument)
If (status < VI_SUCCESS) Then GoTo VisaErrorHandler

'Ask for the instrument's ID.
status = viVPrintf(viInstrument, "*IDN?" + Chr$(10), 0)
If (status < VI_SUCCESS) Then GoTo VisaErrorHandler

'Read the ID as a string.
status = viVScanf(viPNA, "%t", strRes)
If (status < VI_SUCCESS) Then GoTo VisaErrorHandler

'Display the results
MsgBox "Other instrument is: " + strRes
' Close the resource manager session (which closes everything)
Call viClose(defRM)
End

ErrorHandler:
'Display the error message
MsgBox "*** Error : " + Error$, MB_ICONEXCLAMATION
End

VisaErrorHandler:
Dim strVisaErr As String * 200
Call viStatusDesc(defRM, status, strVisaErr)
MsgBox "*** Error : " + strVisaErr

End
End Sub

```

Create a Balanced Measurement using SCPI

This example program does the following:

- creates several Balanced measurements in separate windows
- generates markers
- calculates statistics
- sets limit lines and queries results
- queries a measurement to determine if we have a balanced parameter and what type it is.

Note: By their nature, balanced measurements are extremely sensitive to phase differences between the two RF paths that make up the balanced port, especially at higher frequencies. A good calibration (not performed in this example) is critical to achieving good balanced measurement results.

The SCPI commands in this example are sent over a COM interface using the [SCPIStringParser](#) object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Balanced.vbs. [Learn how to setup and run the macro.](#)

See Other SCPI Example Programs

```
Dim app
Dim scpi
' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
' A comment
scpi.Parse("SYST:FPRESET")
' This example uses DUT topology Bal-Bal -
' a DUT with a balanced input and balanced output.
'
' Port mapping for our DUT:
' logical port 1 = physical ports 1 and 4
' logical port 2 = physical ports 2 and 3
```

```

' The default is:
' logical port 1 = physical ports 1 and 2
' logical port 2 = physical ports 3 and 4
'
' logical 1          logical 2
'
' 1 -----|          |----- 2 +
'          |    DUT   |
' 4 -----|          |----- 3 -

' Turn on Four windows
scpi.Parse("DISP:WIND1:STATE ON")
scpi.Parse("DISP:WIND2:STATE ON")
scpi.Parse("DISP:WIND3:STATE ON")
scpi.Parse("DISP:WIND4:STATE ON")

' Create a trace called "sdd21", and for that trace turn on the balanced
' transformation and set the balanced transformation to BBAL SDD21.
scpi.Parse("CALC:PAR:DEF:EXT ""sdd21"",S11")
scpi.Parse("CALC:PAR:SEL ""sdd21""")
scpi.Parse("CALC:FSIM:BAL:PAR:STATE ON")
scpi.Parse("CALC:FSIM:BAL:PAR:BBAL:DEF SDD21")

' Feed the sdd21 trace to window 1, trace 1
scpi.Parse("DISP:WIND1:TRAC1:FEED ""sdd21""")

' Similarly create 3 more balanced transmission/conversion parameters
' Create Scd21
scpi.Parse("CALC:PAR:DEF:EXT ""scd21"",S11")
scpi.Parse("CALC:PAR:SEL ""scd21""")
scpi.Parse("CALC:FSIM:BAL:PAR:STATE ON")

```

```
scpi.Parse("CALC:FSIM:BAL:PAR:BBAL:DEF SCD21")
scpi.Parse("DISP:WIND1:TRAC2:FEED ""scd21""")
' Create Sdc21
scpi.Parse("CALC:PAR:DEF:EXT ""sdc21"",S11")
scpi.Parse("CALC:PAR:SEL ""sdc21""")
scpi.Parse("CALC:FSIM:BAL:PAR:STATE ON")
scpi.Parse("CALC:FSIM:BAL:PAR:BBAL:DEF SDC21")
scpi.Parse("DISP:WIND1:TRAC3:FEED ""sdc21""")
' Create Scc21
scpi.Parse("CALC:PAR:DEF:EXT ""scc21"",S11")
scpi.Parse("CALC:PAR:SEL ""scc21""")
scpi.Parse("CALC:FSIM:BAL:PAR:STATE ON")
scpi.Parse("CALC:FSIM:BAL:PAR:BBAL:DEF SCC21")
scpi.Parse("DISP:WIND1:TRAC4:FEED ""scc21""")
' Now create logical port 1 reflection parameters, and place them in window 2
scpi.Parse("CALC:PAR:DEF:EXT ""sdd11"",S11")
scpi.Parse("CALC:PAR:SEL ""sdd11""")
scpi.Parse("CALC:FSIM:BAL:PAR:STATE ON")
scpi.Parse("CALC:FSIM:BAL:PAR:BBAL:DEF SDD11")
' Feed the sdd11 trace to window 2, trace 1
scpi.Parse("DISP:WIND2:TRAC1:FEED ""sdd11""")
' Similarly create 3 more balanced reflection/conversion parameters
scpi.Parse("CALC:PAR:DEF:EXT ""scd11"",S11")
scpi.Parse("CALC:PAR:SEL ""scd11""")
scpi.Parse("CALC:FSIM:BAL:PAR:STATE ON")
scpi.Parse("CALC:FSIM:BAL:PAR:BBAL:DEF SCD11")
scpi.Parse("DISP:WIND2:TRAC2:FEED ""scd11""")
scpi.Parse("CALC:PAR:DEF:EXT ""sdc11"",S11")
```

```
scpi.Parse("CALC:PAR:SEL ""sdc11"")
scpi.Parse("CALC:FSIM:BAL:PAR:STATE ON")
scpi.Parse("CALC:FSIM:BAL:PAR:BBAL:DEF SDC11")
scpi.Parse("DISP:WIND2:TRAC3:FEED ""sdc11"")
scpi.Parse("CALC:PAR:DEF:EXT ""scc11",S11")
scpi.Parse("CALC:PAR:SEL ""scc11"")
scpi.Parse("CALC:FSIM:BAL:PAR:STATE ON")
scpi.Parse("CALC:FSIM:BAL:PAR:BBAL:DEF SCC11")
scpi.Parse("DISP:WIND2:TRAC4:FEED ""scc11"")
' Now create reverse transmission parameters, and place them in window 3
scpi.Parse("CALC:PAR:DEF:EXT ""sdd12",S11")
scpi.Parse("CALC:PAR:SEL ""sdd12"")
scpi.Parse("CALC:FSIM:BAL:PAR:STATE ON")
scpi.Parse("CALC:FSIM:BAL:PAR:BBAL:DEF SDD12")
' Feed the sdd11 trace to window 3, trace 1
scpi.Parse("DISP:WIND3:TRAC1:FEED ""sdd12"")
' Similarly create 3 more balanced reverse transmission/conversion parameters
scpi.Parse("CALC:PAR:DEF:EXT ""scd12",S11")
scpi.Parse("CALC:PAR:SEL ""scd12"")
scpi.Parse("CALC:FSIM:BAL:PAR:STATE ON")
scpi.Parse("CALC:FSIM:BAL:PAR:BBAL:DEF SCD12")
scpi.Parse("DISP:WIND3:TRAC2:FEED ""scd12"")
scpi.Parse("CALC:PAR:DEF:EXT ""sdc12",S11")
scpi.Parse("CALC:PAR:SEL ""sdc12"")
scpi.Parse("CALC:FSIM:BAL:PAR:STATE ON")
scpi.Parse("CALC:FSIM:BAL:PAR:BBAL:DEF SDC12")
scpi.Parse("DISP:WIND3:TRAC3:FEED ""sdc12"")
```

```
scpi.Parse("CALC:PAR:DEF:EXT ""scc12"",S11")

scpi.Parse("CALC:PAR:SEL ""scc12"")

scpi.Parse("CALC:FSIM:BAL:PAR:STATE ON")

scpi.Parse("CALC:FSIM:BAL:PAR:BBAL:DEF SCC12")

scpi.Parse("DISP:WIND3:TRAC4:FEED ""scc12"")

' Now create reverse reflection parameters, and place them in window 4

scpi.Parse("CALC:PAR:DEF:EXT ""sdd22"",S11")

scpi.Parse("CALC:PAR:SEL ""sdd22"")

scpi.Parse("CALC:FSIM:BAL:PAR:STATE ON")

scpi.Parse("CALC:FSIM:BAL:PAR:BBAL:DEF SDD22")

' Feed the sdd11 trace to window 3, trace 1

scpi.Parse("DISP:WIND4:TRAC1:FEED ""sdd22"")

' Similarly create 3 more balanced reverse reflection parameters

scpi.Parse("CALC:PAR:DEF:EXT ""scd22"",S11")

scpi.Parse("CALC:PAR:SEL ""scd22"")

scpi.Parse("CALC:FSIM:BAL:PAR:STATE ON")

scpi.Parse("CALC:FSIM:BAL:PAR:BBAL:DEF SCD22")

scpi.Parse("DISP:WIND4:TRAC2:FEED ""scd22"")

scpi.Parse("CALC:PAR:DEF:EXT ""sdc22"",S11")

scpi.Parse("CALC:PAR:SEL ""sdc22"")

scpi.Parse("CALC:FSIM:BAL:PAR:STATE ON")

scpi.Parse("CALC:FSIM:BAL:PAR:BBAL:DEF SDC22")

scpi.Parse("DISP:WIND4:TRAC3:FEED ""sdc22"")

scpi.Parse("CALC:PAR:DEF:EXT ""scc22"",S11")

scpi.Parse("CALC:PAR:SEL ""scc22"")

scpi.Parse("CALC:FSIM:BAL:PAR:STATE ON")

scpi.Parse("CALC:FSIM:BAL:PAR:BBAL:DEF SCC22")

scpi.Parse("DISP:WIND4:TRAC4:FEED ""scc22"")
```

```

scpi.Parse("CALC:FSIM:BAL:DEVIce BBALanced")

scpi.Parse("CALC:FSIM:BAL:TOPology:BBAL:PPORts 1,4,2,3")

' Set up stimulus

scpi.Parse("SENS:SWE:POINts 801")

scpi.Parse("SENS:FREQ:STARt 10e6")

scpi.Parse("SENS:FREQ:STOP 1e9")

' Here we demonstrate how to determine if we have
' a balanced parameter and what type it is.
' Read back one parameter to verify its type

scpi.Parse("CALC:PAR:SEL ""sdd21""")

' Is this a balanced parameter?

isbal = scpi.Parse("CALC:FSIM:BAL:PAR?")

' Which topology/device is set?

device = scpi.Parse("CALC:FSIM:BAL:DEV?")

device = Left( device, Len(device)-1 ) ' strip off newline

' Which parameter are we measuring within that topology?

balparam = scpi.Parse("CALC:FSIM:BAL:PAR:" & device & ":DEF?")

balparam = Left( balparam, Len(balparam)-1 ) ' strip off newline

If isbal Then

WScript.Echo "Balanced Parameter: " & balparam & " in topology: " & device & "."

Else

WScript.Echo "Parameter not balanced."

End If

```

Last Modified:

9-May-2011 Modified to make it work per TS

Create a Measurement using SCPI

This VBScript program creates a new S21 measurement and displays it on the PNA screen.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as NewMeas.vbs. [Learn how to setup and run the macro.](#)

[See Other SCPI Example Programs](#)

```
Dim app
Dim scpi
' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

' A comment
'Preset the analyzer
scpi.Execute ("SYST:FPRreset")
' Create and turn on window 1
scpi.Execute ("DISPlay:WINDow1:STATE ON")
'Define a measurement name, parameter
scpi.Execute ("CALCulate:PARAmeter:DEFine:EXT 'MyMeas',S21")
'Associate ("FEED") the measurement name ('MyMeas') to WINDow (1), and give the new
TRACe a number (1).
scpi.Execute ("DISPlay:WINDow1:TRACe1:FEED 'MyMeas'")
```

Create a Narrowband Point-in-Pulse Measurement using the PNA-X

The following **SCPI** example demonstrates how to create a Narrowband Point-in-Pulse measurement using the Pulsed Application DLL on the [PNA-X](#).

It first gets valid configuration settings and then uses those settings to configure the PNA and internal pulsed generators.

To run this program, you need:

- PNA-X
- [Pulsed Application](#) (Option H08)

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as PulseProfile.vbs. [Learn how to setup and run the macro.](#)

Note: Because of the long length of some commands in this example, word wrapping may occur when copying. These lines require modification after pasting.

See Also

- [Install and register the pulsed .dll](#) on your PC.
- [ConfigEnhancedNB2](#) method for sending and returning parameters to the .dll.
- [ConfigEnhancedNBIFAtten](#) method for setting the receiver IF gain.
- [SCPI IF Configuration](#) commands used in the program.
- [Other Pulse SCPI examples](#)

```
//This example shows you how to perform point in pulse measurement based on
//PNA-X in narrowband mode using SCPI commands.

public partial class Form1 : Form
{
private object pna;
private object scpi;
private Type srvtype;
```

```

private AgilentPNAPulsed.applicationClass pulseApp;

public Form1()
{
InitializeComponent();
}

private string sendScpiCommand(string scpitext)
{
object[] parameter = new object[1];

parameter[0] = scpitext;

return (string)srvttype.InvokeMember("parse", BindingFlags.InvokeMethod, null,
scpi, parameter);
}

private void ConnectToPNA()
{
srvttype = Type.GetTypeFromProgID("AgilentPNA835x.Application", true);

pna = Activator.CreateInstance(srvttype);

scpi = srvttype.InvokeMember("ScpiStringParser", BindingFlags.GetProperty, null,
pna, null);
}

private void NButton_Click(object sender, EventArgs e)
{
double dPRF = 10000, dBW = 500; //PRF=10kHz

double dPhysicalIF = 0, dNCO = 0, dClockFreq = 0;

System.Array aStage1TapArray = null, aStage2TapArray = null, aStage3TapArray =
null;

bool bFixedPRF = true;

double dGateDelay = 0.000002, dGateWidth = 0.000005; //Gate width=50ns

double dSWGGateDelay = 0, dSWGGateWidth = 0;

```

```
int iSWGGateRamp = 0;

double dModPulseWidth = 0.00001;//10 us

double dModPulseDelay = 0;//0us

short myAtten = 0;

pulseApp = new AgilentPNAPulsed.applicationClass();

ConnectToPNA();

//Preset PNA-X

sendScpiCommand("*RST");

//Measure S21

sendScpiCommand("DISP:WIND:TRAC1:DEL");

sendScpiCommand("CALCulate:PARAMeter:DEFine:EXT /'MyMeas/',S21");

sendScpiCommand("DISP:WIND:TRAC1:FEED /'MyMeas/'");

//Set power leveling mode to Openloop

sendScpiCommand("sour:powl:alc:mode open");

//Send desired pulsed parameters to the pulsed configuration DLL.

//The DLL will return a new set of pulse parameters to send to the PNA-X.

pulseApp.ConfigEnhancedNB2(ref dPRF, ref dBW, ref dPhysicalIF, ref dNCO, ref
dClockFreq, ref aStage1TapArray,
ref aStage2TapArray, ref aStage3TapArray, bFixedPRF, dGateDelay, dGateWidth,
ref dSWGGateDelay, ref dSWGGateWidth, ref iSWGGateRamp);

double pulsePeriod = 1 / dPRF;

//Pulse #1 as modulation source

sendScpiCommand("sens:puls:per " + pulsePeriod.ToString()); // 100us

//Set Pulse1 width

sendScpiCommand("sens:puls1:width " + dModPulseWidth.ToString()); //10us

//Set Pulse1 delay

sendScpiCommand("sens:puls1:delay " + dModPulseDelay.ToString()); //10us

//Turn on Pulse1
```

```
sendScpiCommand("SENS:PULS1:STAT 1");

//Set modulation source to Pulse1

sendScpiCommand("sens:path:conf:elem /"PulseModDrive"/,"Pulse1/");

//Enable pulse modulator 1

sendScpiCommand("sens:path:conf:elem /"Src1Out1PulseModEnable"/,"Enable/");

//Pulse #2 controls receiver gate

sendScpiCommand("sens:puls2:width " + dGateWidth.ToString()); //50ns

sendScpiCommand("sens:puls2:delay " + dGateDelay.ToString()); //0

sendScpiCommand("SENS:PULS2:STAT 1");

sendScpiCommand("sens:path:conf:elem /"IFGateA"/,"Pulse2/");

sendScpiCommand("sens:path:conf:elem /"IFGateB"/,"Pulse2/");

sendScpiCommand("sens:path:conf:elem /"IFGateR1"/,"Pulse2/");

sendScpiCommand("sens:path:conf:elem /"IFGateR2"/,"Pulse2/");

//Set IFBW

sendScpiCommand("SENS:BWID " + dBW.ToString());

//Configure IF path

sendScpiCommand("sens:path:conf:elem /"IFSigPathAll"/,"NBF/");

sendScpiCommand("SENS:IF:FILT:AUTO 0");

sendScpiCommand("SENS:IF:FREQ:AUTO 0");

sendScpiCommand("SENS:IF:FREQ " + dPhysicalIF.ToString());

// Set filter stages based on pulse parameters

sendScpiCommand("SENS:IF:FILT:STAGE1:FREQ " + dNCO.ToString());

//Convert Stage1TapArray to string

string buf1 = new string(' ', 1000);

for (int i = 0; i < aStage1TapArray.GetLength(0); i++)

{

buf1 = buf1 + aStage1TapArray.GetValue(i).ToString() + ",";

}

}
```

```

buf1 = buf1.Trim();

buf1 = buf1.Substring(0, buf1.Length - 1);

//Convert Stage1TapArray to string

string buf2 = new string(' ', 1000);

for (int j = 0; j < aStage2TapArray.GetLength(0); j++)

{

buf2 = buf2 + aStage2TapArray.GetValue(j).ToString() + ",";

}

buf2 = buf2.Trim();

buf2 = buf2.Substring(0, buf2.Length - 1);

//Set IF Filter Stage1 and Stage2 Coeficent

sendScpiCommand("SENS:IF:FILT:STAG1:COEF " + buf1);

sendScpiCommand("SENS:IF:FILT:STAG2:COEF " + buf2);

if (dSWGateWidth == 0) // No valid SW gate

{

sendScpiCommand("SENS:IF:FILT:STAG3:TYPE 'RECT'");

sendScpiCommand("SENS:IF:FILT:STAG3:PAR 'C'," +
aStage3TapArray.GetValue(0).ToString());

sendScpiCommand("SENS:PULS0:STAT 0");

}

else

{

sendScpiCommand("SENS:IF:FILT:STAG3:TYPE 'PWIN'");

sendScpiCommand("SENS:IF:FILT:STAG3:PAR 'C'," +
aStage3TapArray.GetValue(0).ToString());

sendScpiCommand("SENS:IF:FILT:STAG3:PAR 'P'," + pulsePeriod.ToString());

sendScpiCommand("SENS:IF:FILT:STAG3:PAR 'D'," + dSWGateDelay.ToString());

sendScpiCommand("SENS:IF:FILT:STAG3:PAR 'W'," + dSWGateWidth.ToString());

sendScpiCommand("SENS:IF:FILT:STAG3:PAR 'R'," + iSWGateRamp.ToString());

```

```

double pulse0Width = 1 / dClockFreq;

sendScpiCommand("sens:puls0:width " + pulse0Width.ToString());

sendScpiCommand("sens:puls0:delay 0");

sendScpiCommand("SENS:PULS0:STAT 1");

}

pulseApp.ConfigEnhancedNBIFatten(dPRF, dGateWidth, ref myAtten);

sendScpiCommand("sens:path:conf:elem /"NBIFATNA"/," + myAtten.ToString() +
"/");

sendScpiCommand("sens:path:conf:elem /"NBIFATNB"/," + myAtten.ToString() +
"/");

sendScpiCommand("sens:path:conf:elem /"NBIFATNR1"/," + myAtten.ToString() +
"/");

sendScpiCommand("sens:path:conf:elem /"NBIFATNR2"/," + myAtten.ToString() +
"/");

//Set start and stop frequency

sendScpiCommand("SENS:FREQ:STAR 1000000000");

sendScpiCommand("SENS:FREQ:STOP 2000000000");

//Single Sweep

sendScpiCommand("SENS:SWE:MODE SING");

sendScpiCommand("DISP:WIND:TRAC:Y:AUTO");

}

}

```

Last Modified:

2-Jul-2008 New topic.

Create a Narrowband Pulse Profile Measurement using the PNA-X

The following **SCPI** example demonstrates how to create a Narrowband Pulse Profile measurement using the Pulsed Application DLL on the [PNA-X](#).

It first gets valid configuration settings and then uses those settings to configure the PNA and internal pulsed generators.

To run this program, you need:

- PNA-X
- [Pulsed Application](#) (Option H08)

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as PulseProfile.vbs. [Learn how to setup and run the macro.](#)

Note: Because of the long length of some commands in this example, word wrapping may occur when copying. These lines require modification after pasting.

See Also

- [Install and register the pulsed .dll](#) on your PC.
- [ConfigEnhancedNB2](#) method for sending and returning parameters to the .dll.
- [ConfigEnhancedNBIFAtten](#) method for setting the receiver IF gain.
- [SCPI IF Configuration](#) commands used in the program.
- [Other Pulse SCPI examples](#)

```
public partial class Form1 : Form
{
private object pna;
private object scpi;
private Type srvtype;
private AgilentPNAPulsed.applicationClass pulseApp;
public Form1()
{
```

```

InitializeComponent();

}

private string sendScpiCommand(string scpitext)

{

object[] parameter = new object[1];

parameter[0] = scpitext;

return (string)srvttype.InvokeMember("parse", BindingFlags.InvokeMethod, null,
scpi, parameter);

}

private void ConnectToPNA()

{

srvttype = Type.GetTypeFromProgID("AgilentPNA835x.Application", true);

pna = Activator.CreateInstance(srvttype);

scpi = srvttype.InvokeMember("ScpiStringParser", BindingFlags.GetProperty, null,
pna, null);

}

private void NBGPIBbutton_Click(object sender, EventArgs e)

{

double dPRF = 10000, dBW = 500; //PRF=10kHz

double dPhysicalIF = 0, dNCO = 0, dClockFreq = 0;

System.Array aStage1TapArray = null, aStage2TapArray = null, aStage3TapArray =
null;

bool bFixedPRF = true;

double dGateDelay = 0.0, dGateWidth = 0.00000005; //Gate width=50ns

double dSWGateDelay = 0, dSWGateWidth = 0;

int iSWGateRamp = 0;

double dModPulseWidth = 0.00001;//10 us

double dModPulseDelay = 0.00001;//10us

short myAtten = 0;

```

```

pulseApp = new AgilentPNAPulsed.applicationClass();

ConnectToPNA();

//Preset PNA-X

sendScpiCommand("*RST");

//Measure S21

sendScpiCommand("DISP:WIND:TRAC1:DEL");

sendScpiCommand("CALCulate:PARAMeter:DEFine:EXT /'MyMeas/',S21");

sendScpiCommand("DISP:WIND:TRAC1:FEED /'MyMeas/'");

//Set power leveling mode to Openloop

sendScpiCommand("sour:pow1:alc:mode open");

//Send desired pulsed parameters to the pulsed configuration DLL.

//The DLL will return a new set of pulse parameters to send to the PNA-X.

pulseApp.ConfigEnhancedNB2(ref dPRF, ref dBW, ref dPhysicalIF, ref dNCO, ref
dClockFreq, ref aStage1TapArray,

ref aStage2TapArray, ref aStage3TapArray, bFixedPRF, dGateDelay, dGateWidth,

ref dSWGateDelay, ref dSWGateWidth, ref iSWGateRamp);

double pulsePeriod = 1 / dPRF;

//Pulse #1 as modulation source

sendScpiCommand("sens:puls:per " + pulsePeriod.ToString()); // 100us

//Set Pulse1 width

sendScpiCommand("sens:puls1:width "+dModPulseWidth.ToString()); //10us

//Set Pulse1 delay

sendScpiCommand("sens:puls1:delay " + dModPulseDelay.ToString()); //10us

//Turn on Pulse1

sendScpiCommand("SENS:PULS1:STAT 1");

//Set modulation source to Pulse1

sendScpiCommand("sens:path:conf:elem /"PulseModDrive"/,"Pulse1/");

```

```
//Enable pulse modulator 1

sendScpiCommand("sens:path:conf:elem /"Src1Out1PulseModEnable"/,"/Enable/");

//Pulse #2 controls receiver gate

sendScpiCommand("sens:puls2:width " + dGateWidth.ToString()); //50ns

sendScpiCommand("sens:puls2:delay " + dGateDelay.ToString()); //0

sendScpiCommand("SENS:PULS2:STAT 1");

sendScpiCommand("sens:path:conf:elem /"IFGateA"/,"/Pulse2/");

sendScpiCommand("sens:path:conf:elem /"IFGateB"/,"/Pulse2/");

sendScpiCommand("sens:path:conf:elem /"IFGateR1"/,"/Pulse2/");

sendScpiCommand("sens:path:conf:elem /"IFGateR2"/,"/Pulse2/");

//Set IFBW

sendScpiCommand("SENS:BWID "+dBW.ToString());

//Configure IF path

sendScpiCommand("sens:path:conf:elem /"IFSigPathAll"/,"/NBF/");

sendScpiCommand("SENS:IF:FILT:AUTO 0");

sendScpiCommand("SENS:IF:FREQ:AUTO 0");

sendScpiCommand("SENS:IF:FREQ "+dPhysicalIF.ToString());

// Set filter stages based on pulse parameters

sendScpiCommand("SENS:IF:FILT:STAGE1:FREQ " + dNCO.ToString());

//Convert Stage1TapArray to string

string buf1 = new string(' ', 1000);

for (int i = 0; i < aStage1TapArray.GetLength(0); i++)

{

buf1 = buf1 + aStage1TapArray.GetValue(i).ToString()+",";

}

buf1 = buf1.Trim();

buf1 = buf1.Substring(0, buf1.Length - 1);

//Convert Stage1TapArray to string
```

```

string buf2 = new string(' ', 1000);

for (int j = 0; j < aStage2TapArray.GetLength(0); j++)
{
buf2 = buf2 + aStage2TapArray.GetValue(j).ToString() + ",";
}

buf2 = buf2.Trim();

buf2 = buf2.Substring(0, buf2.Length - 1);

//Set IF Filter Stage1 and Stage2 Coeficent

sendScpiCommand("SENS:IF:FILT:STAG1:COEF " + buf1 );

sendScpiCommand("SENS:IF:FILT:STAG2:COEF " + buf2);

if (dSWGateWidth == 0) // No valid SW gate
{

sendScpiCommand("SENS:IF:FILT:STAG3:TYPE 'RECT'");

sendScpiCommand("SENS:IF:FILT:STAGe3:PAR 'C'," +
aStage3TapArray.GetValue(0).ToString());

sendScpiCommand("SENS:PULS0:STAT 0");

}

else

{

sendScpiCommand("SENS:IF:FILT:STAGe3:TYPE 'PWIN'");

sendScpiCommand("SENS:IF:FILT:STAGe3:PAR 'C'," +
aStage3TapArray.GetValue(0).ToString());

sendScpiCommand("SENS:IF:FILT:STAGe3:PAR 'P'," + pulsePeriod.ToString());

sendScpiCommand("SENS:IF:FILT:STAGe3:PAR 'D'," + dSWGateDelay.ToString());

sendScpiCommand("SENS:IF:FILT:STAGe3:PAR 'W'," + dSWGateWidth.ToString());

sendScpiCommand("SENS:IF:FILT:STAGe3:PAR 'R'," + iSWGateRamp.ToString());

double pulse0Width=1/dClockFreq;

sendScpiCommand("sens:puls0:width " + pulse0Width.ToString());

sendScpiCommand("sens:puls0:delay 0");
}
}

```

```
sendScpiCommand("SENS:PULS0:STAT 1");
}

pulseApp.ConfigEnhancedNBIFatten(dPRF, dGateWidth, ref myAtten);
sendScpiCommand("sens:path:conf:elem /"NBFATNA/",/" "+myAtten.ToString()+ "/"");
sendScpiCommand("sens:path:conf:elem /"NBFATNB/",/" " + myAtten.ToString() +
"/"");
sendScpiCommand("sens:path:conf:elem /"NBFATNR1/",/" " + myAtten.ToString() +
"/"");
sendScpiCommand("sens:path:conf:elem /"NBFATNR2/",/" " + myAtten.ToString() +
"/"");

//Run pulse profile using below several lines

//Set CW Mode
sendScpiCommand("SENS:SWE:TYPE CW");
double startTime, stopTime, stepTime;
int myProfilePoints;
startTime = 0.00001;//10us
stopTime = 0.00005;//50us
stepTime = 0.00000005;//50ns
myProfilePoints = (int)((stopTime - startTime) / stepTime) + 1;
sendScpiCommand("SENS:SWE:POIN " + myProfilePoints.ToString());

// Test gates
sendScpiCommand("sens:puls1:delay " + startTime.ToString());
sendScpiCommand("SENS:PULS2:DINC " + stepTime.ToString());

//Single Sweep
sendScpiCommand("SENS:SWE:MODE SING");
sendScpiCommand("DISP:WIND:TRAC:Y:AUTO");
}
}
```

Last Modified:

2-Jul-2008 New topic.

Configure a PMAR Device

This VB Script program configures a new Power Meter as Receiver device and creates a trace using the PMAR.

[Learn more about Power Meter as a Receiver](#)

These programs can be run as a macro in the PNA. To do this, copy the code into a text editor file such as Notepad and save on the PNA hard drive as PMAR.vbs. [Learn how to setup and run the macro.](#)

[See all External Device Configuration commands](#)

See Other SCPI Example Programs

```
' This section gets the PNA application
' starts the scpi parser, and presets the PNA

dim app
Set app = CreateObject("AgilentPNA835x.Application")
set scpi = app.ScpiStringParser
scpi.parse "*rst"
scpi.parse "Syst:conf:edev:add 'newpmar1'"
scpi.parse "Syst:conf:edev:dtype 'newpmar1', 'Power Meter'"
scpi.parse "Syst:conf:edev:ioconfig 'newpmar1', 'gpib0::14::instr'"
scpi.parse "Syst:conf:edev:pmar:sens 'newpmar1', 1"
scpi.parse "Syst:conf:edev:pmar:read:count 'newpmar1', 10"
scpi.parse "Syst:conf:edev:pmar:read:ntolerance 'newpmar1', 0.1"
scpi.parse "Syst:conf:edev:pmar:sens 'newpmar1', 1"
scpi.parse "Syst:conf:edev:pmar:fmin 'newpmar1', 100000000"
scpi.parse "Syst:conf:edev:pmar:fmax 'newpmar1', 100000000000"
scpi.parse "Syst:conf:edev:pmar:flim 'newpmar1', 0"
scpi.parse "Syst:conf:edev:pmar:tabl:rfac 'newpmar1', 100"
scpi.parse "Syst:conf:edev:pmar:tabl:cfac:freq 'newpmar1', 1e9, 2e9, 3e9"
```

```
scpi.parse "Syst:conf:edev:pmar:tabl:cfac:data 'newpmar1', 99, 98, 99"
scpi.parse "Syst:conf:edev:pmar:tabl:loss:stat 'newpmar1', 1"
scpi.parse "Syst:conf:edev:pmar:tabl:loss:freq 'newpmar1', 1e9, 2e9, 3e9"
scpi.parse "Syst:conf:edev:pmar:tabl:loss:data 'newpmar1', -1, -2, -3"

'Activate and enable the PMAR external device

scpi.parse "Syst:conf:edev:ioen 'newpmar1', 1"
scpi.parse "Syst:conf:edev:stat 'newpmar1', 1"

'Create a PMAR trace with power meter connected to port 3

'Use Calc:Par:Def:Ext - NOT CALC:PAR:DEF!!

scpi.parse "CALC:PAR:DEF:EXT 'myPMARTrace', 'newpmar1,3'"
```

Last Modified:

11-Feb-2011	Renamed
3-Sep-2009	MX New topic

Create a Swept IMDX Measurement

This program configures several Swept IMDx parameters using power sweep. In this configuration, tone power is swept from -20 dBm to -5 dBm while the Input, LO, and Output frequencies are fixed as follows:

- Input center freq= 2.50 GHz
- Tone Delta freq = 10 MHz (f1 = 2.495 GHz and f2 = 2.505 GHz).
- LO freq = 2.00 GHz
- Output freq = 4.50 GHz

This program also allows you to optionally load a .mxr file to perform mixer setup.

To run this program without error, an external source named 'PSG' must be connected to drive the LO.

This program can be run as a macro in the PNA. To do this, copy the code into a text editor file such as Notepad and save on the PNA hard drive as IMDX.vbs. [Learn how to setup and run the macro.](#)

[See all Swept IMD commands.](#)

[See all Mixer Setup commands.](#)

See Other SCPI Example Programs

```
Dim app
Dim scpi
Dim err
'
'Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
'
'Preset the system
scpi.parse "SYST:FPR"
```

```
scpi.parse "DISP:WIND1:STAT ON"
'
'Create an IMDX measurement in Channel 1 and display it as trace 1
in window 1
scpi.parse "CALC1:CUST:DEF 'ch1IMDX', 'Swept IMD Converters',
'PwrMain'"
scpi.parse "DISP:WIND1:TRAC1:FEED 'ch1IMDX'"
'
'Put the channel in trigger hold
scpi.parse "SENS:SWE:MODE HOLD"
scpi.parse "SENS:IMD:SWE:TYPE POW"
'Put the channel in trigger hold and setup all the mixer
parameters
scpi.parse "SENS1:SWE:MODE HOLD"
scpi.parse "SENS1:IMD:SWE:TYPE POW"
scpi.parse "SENS1:MIX:INP:FREQ:MODE FIXED"
scpi.parse "SENS1:MIX:LO:FREQ:MODE FIXED"
scpi.parse "SENS1:MIX:OUTP:FREQ:MODE FIXED"
scpi.parse "SENS1:MIX:INP:FREQ:FIX 2500000000"
scpi.parse "SENS1:MIX:LO:FREQ:FIX 2000000000"
scpi.parse "SENS1:MIX:OUTP:FREQ:SID HIGH"
scpi.parse "SENS1:MIX:CALC OUTP"
scpi.Parse "SENS:MIX:APPLY"
'First apply the settings, then set LO Name
scpi.Parse "SENS:MIX:LO:NAME 'PSG'"
scpi.parse "SENS1:MIX:LO:POW 10"
scpi.Parse "SENS:MIX:APPLY"
```

```
'Optionally, put the channel in hold and load an
'existing .mxr file with all the mixer settings
'scpi.parse "SENS1:SWE:MODE HOLD"
'scpi.parse "SENS1:MIX:LOAD 'c:\users\public\network
analyzer\documents\Mixer\IMD\Ch1.mxr'"
'
'Make additional IMD settings
scpi.parse "SENS1:IMD:TPOW:COUP:STAT ON"
scpi.parse "SENS1:IMD:TPOW:F1:STAR -20"
scpi.parse "SENS1:IMD:TPOW:F1:STOP -5"
scpi.parse "SENS1:IMD:FREQ:DFR:CW 10000000"
scpi.parse "SENS1:SWE:POIN 201"
scpi.parse "SENS1:imd:ifbw:main 1000"
scpi.parse "SENS1:imd:ifbw:imt 500"
scpi.parse "SOUR1:POW2:AMPL -5"
'
'Create additional measurements in the channel
scpi.parse "CALC1:CUST:DEF 'ch1IMDX2', 'Swept IMD Converters',
'IM3'"
scpi.parse "DISP:WIND1:TRAC2:FEED 'ch1IMDX2'"
scpi.parse "CALC1:CUST:DEF 'ch1IMDX3', 'Swept IMD Converters',
'OIP3'"
scpi.parse "DISP:WIND1:TRAC3:FEED 'ch1IMDX3'"
scpi.parse "CALC1:CUST:DEF 'ch1IMDX4', 'Swept IMD Converters',
'IIP3'"
scpi.parse "DISP:WIND1:TRAC4:FEED 'ch1IMDX4'"
scpi.parse "CALC1:CUST:DEF 'ch1IMDX5', 'Swept IMD Converters',
```

```
'ToneGain' "
```

```
scpi.parse "DISP:WIND1:TRAC5:FEED 'ch1IMDX5' "
```

```
'
```

```
'Take a single sweep to apply all stimulus changes
```

```
scpi.parse "*cls;*ese 1"
```

```
scpi.parse "sens1:swe:mode SING;*OPC?"
```

```
'
```

```
'Check for errors
```

```
err=scpi.parse ("SYST:ERR?")
```

```
MsgBox(err)
```

Last Modified:

8-Mar-2012	Updated LO name order (ZW)
21-Sep-2010	Fixed error
15-Jul-2010	MX New topic

Create a Wideband Pulsed Measurement using the PNA-X

This Visual Basic example shows you how to configure the PNA-X internal pulse generators and modulators to make wideband pulsed measurements in **pulse profile** mode using the PNA-X.

[Visit the PNA website](#) where you can download a free Wideband Pulsed Application that performs this measurement on the PNA-X.

[See all SCPI Pulsed examples](#)

```
Private Sub Form_Load()  
  
Dim app  
  
Dim scpi  
  
    ' Create / Get the PNA application.  
  
    Set app = CreateObject("AgilentPNA835x.Application")  
  
    Set scpi = app.ScpiStringParser  
  
    'Preset the analyzer  
  
    scpi.Execute ("*RST")  
  
  
    'Set BW to 5 MHz  
  
    scpi.Execute ("SENS:BWID 5MHZ")  
  
    'Set sweep type to CW mode  
  
    scpi.Execute ("SENS:SWE:TYPE CW")  
  
    'Delete S11 trace  
  
    scpi.Execute ("DISP:WIND:TRAC1:DEL")  
  
    'Create S21 trace  
  
    scpi.Execute ("CALC:PAR:DEF:EXT 'MyMeas',S21")  
  
    scpi.Execute ("DISP:WIND:TRAC1:FEED 'MyMeas'")  
  
    'Set modulation source to Pulse1  
  
    scpi.Execute ("sens:path:conf:elem 'PulseModDrive','Pulse1'")  
  
    'Set power leveling mode to Openloop
```

```
scpi.Execute ("sour:pow1:alc:mode open")

'Enable pulse modulator 1

scpi.Execute ("sens:path:conf:elem 'Src1Out1PulseModEnable','Enable'")

'Set clock of internal pulse generator to internal

scpi.Execute ("sens:path:conf:elem 'PulseTrigInput','Internal'")

'Turn on Pulse0

scpi.Execute ("SENS:PULS0:STAT 1")

'Turn on Pulse1

scpi.Execute ("SENS:PULS1:STAT 1")

'Set pulse period to 1 ms

scpi.Execute ("sens:puls:per .001")

'Set Pulse1 width to 10 us

scpi.Execute ("sens:puls1:width 0.00001")

'Set Pulse1 delay to 8 us

scpi.Execute ("sens:puls1:delay 0.000008")

'Set Pulse0 width to 1 us

scpi.Execute ("sens:puls0:width 0.000001")

'Set Pulse0 delay to 400 ns

scpi.Execute ("sens:puls0:delay 0.0000004")

'Set trigger scope to Channel

scpi.Execute ("TRIG:SCOP CURRENT")
```

End Sub

Last Modified:

2-Jul-2008 New topic.

Create an FOM Measurement

All three VBScript examples in this topic create a FOM measurement with the following attributes:

- Sweep the Source (input) from 1 GHz to 2 GHz
- Sweep the Receivers (output) from 2 GHz to 3 GHz
- You provide an LO at 1 GHz

[Learn more about Frequency Offset Mode](#)

These programs can be run as a macro in the PNA. To do this, copy the code into a text editor file such as Notepad and save on the PNA hard drive as FOM.vbs. [Learn how to setup and run the macro.](#)

[See Other SCPI Example Programs](#)

The following example will run on any PNA model with FOM (opt 080). However, these commands have no provisions for internal second source. It uses [Sens:Offset commands](#) introduced before 'enhanced FOM' was released for the A.07.10 release.

```
' This section gets the PNA application
' starts the scpi parser, and presets the PNA
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
scpi.Execute("SYST:FPRESET")
' Create and turn on window 1
scpi.Execute ("DISPlay:WINDow1:STATE ON")
'Define a measurement name, parameter
scpi.Execute ("CALCulate:PARAMeter:DEFine:EXT 'MyMeas',S21")
'Associate ("FEED") the measurement name ('MyMeas') to WINDow (1)
'and give the new TRACE a number (1).
scpi.Execute ("DISPlay:WINDow1:TRACe1:FEED 'MyMeas'")
```

```
scpi.Execute ("SENS:FREQ:START 1e9")
scpi.Execute ("SENS:FREQ:STOP 2e9")
'set the receivers to be 2e9 -> 3e9
scpi.Execute ("SENS:OFFS:OFFS 1e9")
scpi.Execute ("SENS:OFFS ON")
```

The following example can be run ONLY on a PNA with revision A.07.10 or later and has FOM (opt 080). It uses new [Sens:FOM commands](#).

```
' This section gets the PNA application
' starts the scpi parser, and presets the PNA
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
scpi.Execute("SYST:FPRESET")
' Create and turn on window 1
scpi.Execute ("DISPlay:WINDow1:STATE ON")
'Define a measurement name, parameter
scpi.Execute ("CALCulate:PARAMeter:DEFine 'MyMeas',S21")
'Associate ("FEED") the measurement name ('MyMeas') to WINDow (1),
and give the new TRACe a number (1).
scpi.Execute ("DISPlay:WINDow1:TRACe1:FEED 'MyMeas'")

scpi.Execute("SENS:FREQ:START 1e9")
scpi.Execute("SENS:FREQ:STOP 2e9")
'set the receivers to be 2e9 -> 3e9
scpi.Execute("SENS:FOM:RANG3:FREQ:OFFS 1e9")
scpi.Execute("SENS:OFFS ON")
```

The following example can be run ONLY on a PNA with a second internal source, has revision A.07.10 or later, and has FOM (opt 080). It uses the internal 2nd source for the fixed LO frequency.

```
' This section gets the PNA application
' starts the scpi parser, and presets the PNA
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
scpi.Execute("SYST:FPRESET")
' Create and turn on window 1
scpi.Execute ("DISPlay:WINDow1:STATE ON")
'Define a measurement name, parameter
scpi.Execute ("CALCulate:PARAMeter:DEFine 'MyMeas',S21")
'Associate ("FEED") the measurement name ('MyMeas') to WINDow (1)
'and give the new TRACe a number (1).
scpi.Execute ("DISPlay:WINDow1:TRACe1:FEED 'MyMeas'")

scpi.Execute ("SENS:FREQ:START 1e9")
scpi.Execute ("SENS:FREQ:STOP 2e9")
'set the receivers to be 2e9 -> 3e9
scpi.Execute ("SENS:FOM:RANG3:FREQ:OFFS 1e9")
'setup the 2nd source frequencies
scpi.Execute ("SENS:FOM:RANG4:COUP 0")
scpi.Execute ("SENS:FOM:RANG4:FREQ:START 1e9")
scpi.Execute ("SENS:FOM:RANG4:FREQ:STOP 1e9")
'turn off coupling
scpi.Execute ("SOUR:POW:COUP 0")
'set LO power to 10dBm
```

```
scpi.Execute ("SOUR:POW3 10")  
  
'turn ON port 3, our LO signal  
  
scpi.Execute ("SOUR:POW3:MODE ON")  
  
scpi.Execute ("SENS:FOM:STAT ON")
```

Last Modified:

9-Oct-2007 MX New topic

Create an iTMSA Measurement

This example program does the following:

- Create an iTMSA Balanced Sdd21 measurement
- Set sweep type = power
- Set phase offset on balanced port 1=180 degrees

The SCPI commands in this example are sent over a COM interface using the [SCPIStringParser](#) object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Balanced.vbs. [Learn how to setup and run the macro.](#)

See Other SCPI Example Programs

```
Dim app
Dim scpi
' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
' Reset the system
scpi.Parse("SYST:FPRESET")
' This example uses DUT topology Bal-Bal -
' a DUT with a True-mode balanced input and balanced output.
'
' Port mapping for our DUT:
' logical port 1 = physical ports 1 and 3
' logical port 2 = physical ports 2 and 4
'
'   logical 1           logical 2
'
' 1 -----|           |----- 2 +
'           |           |
'           |   DUT   |
'           |           |
' 3 -----|           |----- 4 -
```

```
' Turn on a window
scpi.Parse("DISP:WIND1:STATE ON")

' Create a trace called "sdd21", and for that trace turn on the balanced
' transformation and set the balanced transformation to BBAL SDD21.
scpi.Parse("CALC:PAR:DEF:EXT ""sdd21"",S11")

' Feed the sdd21 trace to window 1, trace 1
scpi.Parse("DISP:WIND1:TRAC1:FEED ""sdd21""")
scpi.Parse("CALC:PAR:SEL ""sdd21""")

' Set the topology of measurement
scpi.Parse("CALC:FSIM:BAL:DEVIce BBALanced")
scpi.Parse("CALC:FSIM:BAL:TOPology:BBAL:PPORTs 1,3,2,4")

' Set up stimulus
scpi.Parse("SENS:SWE:POINTs 801")
scpi.Parse("SENS:FREQ:START 10e6")
scpi.Parse("SENS:FREQ:STOP 1e9")
scpi.Parse("CALC:FSIM:BAL:PAR:STATE ON")
scpi.Parse("CALC:FSIM:BAL:PAR:BBAL:DEF SDD21")

' Recall a 4-port Cal Set or perform a 4-port Cal here

' Set the sweep type to power sweep
scpi.Parse("SENS:SWE:TYPE POWER")

' Set iTMSA parameters
scpi.Parse("CALC:FSIM:BAL:BPOR1:OFFS:PHAS 180")
scpi.Parse("CALC:FSIM:BAL:FIXT:OFFS:PHAS ON")
scpi.Parse("CALC:FSIM:BAL:STIM:MODE TM")
```

Create an SMC Fixed Output Measurement

This VB Script example creates a calibrated SMC fixed output measurement using an external, controlled LO. Then a single sweep is taken and data is retrieved. The external LO is NOT required when using the internal second PNA source for the LO.

Requirements:

- The external LO should be configured to match the SENS:MIX:LO:NAME command below.

Fixed output measurements require that an external LO source be swept and synchronized with the PNA source. FCA performs this synchronization using the external source configuration settings. See [Configure an External Source](#) using SCPI.

The fastest, and recommended, method of controlling the LO source is [Hardware List \(BNC\) triggering mode](#). However, in this mode, FCA channels will not respond to manual triggers. Therefore, the example uses the following mechanism to trigger a sweep:

```
Write "SENS:SWE:MODE HOLD"      'place channel 1 in HOLD mode
Write "INIT:CONT ON"             'place PNA in internal trigger mode
Write "SENS:SWE:MODE SINGLE"
Write "*OPC?"                    'wait until the sweep is complete
Read
```

The SCPI commands in this example are sent over a COM interface using the [SCPIStringParser](#) object. You can run a VBScript (*.vbs) program from the PNA [using Macros](#). To run this program, copy the following code into a text editor and save it as a *.vbs file.

```
option explicit

' Setup infrastructure to use the SCPI over COM

dim app

set app = createobject("Agilentpna835x.application")

dim p

set p = app.scpistringparser

dim returnStr

sub Write (command)

    if len(returnStr) <> 0 then

        err.Raise 55,"Write","Query Unterminated"
```

```

end if

returnStr = p.parse(command)

end sub

sub WriteIgnoreError(command)

returnStr = p.Execute(command)

p.Parse("SYST:ERR?") ' clear error queue

end sub

function Read

if len(returnStr) = 0 then

err.Raise 55,"Read","Bad read"

end if

Read = returnStr

returnStr = ""

end function

Write "SYST:PRES"

' When programming in remote mode, hold mode is recommended

Write "SENS:SWE:MODE HOLD"

' Delete the standard measurement

Write "CALC:PAR:DEL:ALL"

' Create an SC21 measurement

Write "CALC:CUST:DEF 'MySC21', 'Scalar Mixer/Converter', 'SC21'"

Write "DISP:WIND:TRACE:FEED 'MySC21'"

Write "CALC:PAR:SEL 'MySC21'"

' Set number of points to 11

Write "SENS:SWE:POIN 11"

' Setup the mixer parameters for a swept LO, fixed output measurement

Write "SENS:MIX:INP:FREQ:START 200e6"

Write "SENS:MIX:INP:FREQ:STOP 700e6"

```

```
Write "SENS:MIX:LO:FREQ:MODE Swept"

Write "SENS:MIX:OUTPUT:FREQ:FIX 3.4e9"

Write "SENS:MIX:OUTP:FREQ:SID HIGH"

Write "SENS:MIX:CALC LO_1"

Write "SENS:MIX:INP:POW -17"

Write "SENS:MIX:LO:POW 10"

Write "SENS:MIX:APPLY

' Specify the LO name, for controlled LO.

' This name is setup in the External Source Config Dialog

Write "SENS:MIX:LO:NAME '8360'"

Write "SENS:MIX:APPLY

' Create an S11 in the same channel

Write "CALC:CUST:DEF 'MyS11', 'Scalar Mixer/Converter', 'S11'"

Write "DISP:WIND:TRACE2:FEED 'MyS11'"

Write "CALC:PAR:SEL 'MyS11'"

' Create an IPwr in the same channel

Write "CALC:CUST:DEF 'MyIPwr', 'Scalar Mixer/Converter', 'IPwr'"

Write "DISP:WIND:TRACE3:FEED 'MyIPwr'"

Write "CALC:PAR:SEL 'MyIPwr'"

' Create an OPwr in the same channel

Write "CALC:CUST:DEF 'MyOPwr', 'Scalar Mixer/Converter', 'OPwr'"

Write "CALC:PAR:SEL 'MyOPwr'"

Write "DISP:WIND:TRACE4:FEED 'MyOPwr'"

' Perform a single sweep, synchronously. When *OPC returns, the sweep is done

Write "SENS:SWE:MODE SINGLE"

Write "*OPC?"

Read

' Retrieve the SC21 data
```

```
Write "CALC:PAR:SEL 'MySC21'"
Write "CALC:DATA? SDATA"
dim data
data = Read()
wscript.echo("SC21=" & data)
'Retrieve the S11 data
Write "CALC:PAR:SEL 'MyS11'"
Write "CALC:DATA? SDATA"
data = Read()
wscript.echo("S11=" & data)
```

Last Modified:

8-Mar-2011 Updated with new SMC commands A.09.33.

Create and Cal a GCA Measurement

This VBS program does the following:

- creates and configures GCA to perform a SMART Sweep
- performs a calibration using an ECal with 3.5 mm Female on Port A and 3.5 mm Male connectors on Port B

This program can be run as a macro in the PNA. To do this, copy the code into a text editor file such as Notepad and save on the PNA hard drive as GCA.vbs. Learn how to setup and run the macro.

See the Gain Compression commands

```
option explicit
```

```
dim CompLevel , Tolerance , StartFreq , StopFreq , NumFreqs , Scale ,  
LinearPower
```

```
dim AcqMode , BackOff , StartPower , StopPower , NumPowers ,  
EnableInterp , CompAlg
```

```
dim DwellTime , IFBandwidth , ShowIterations , host , app , parser
```

```
' GCA Settings/Values
```

```
''
```

```
' Acquisition Mode:
```

```
' naSmartSweep = 0
```

```
' naSweepPowerAtEachFreq2D = 1
```

```
' naSweepFreqAtEachPower2D = 2
```

```
''
```

```
' Compression Algorithm
```

```
' naCompressionFromLinearGain = 0
```

```
' naCompressionFromMaximumGain = 1
```

```
' naBackoffCompression = 2
```

```
' naXYCompression = 3
```

```

''
'' EndOfSweepOperation
'' naDefaultPowerSet = 0
'' naSetToStartPower = 1
'' naSetToStopPower = 2
'' naSetRFOff = 3
''

CompLevel          = 1          ' 1 dB compression level
Tolerance          = 0.05      ' SMART Sweep tolerance
StartFreq          = 1E9
StopFreq           = 9E9
NumFreqs           = 201
Scale              = 0.1
LinearPower        = -20
BackOff            = 10        ' Not used for Deviation from linear gain
StartPower         = -20
StopPower          = 8
NumPowers          = 60        ' Not used for SMART Sweep
DwellTime          = 0.0005    ' Allow some time for DUT bias/thermal
effects
IFBandwidth        = 1000      ' Reasonable trace noise at -20 dBm
EnableInterp       = False     ' Disable interpolation
AcqMode            = 0         ' Smart Sweep
CompAlg            = 0         ' Deviation from linear gain
ShowIterations     = False     ' Configure SMART to not show iteration
results
dim objargs

```

```

set objargs = wscript . Arguments
if ( objArgs . Count = 1) then host = objargs (0)
.....
'' Create and Configuration GCA Channel:
.....

set app = CreateObject ("Agilentpna835x.application" )
set parser = app .ScpiStringParser
call SetupGCA ( parser ,_
                StartFreq ,_
                StopFreq ,_
                NumFreqs ,_
                EnableInterp ,_
                Scale ,_
                CompLevel ,_
                LinearPower ,_
                AcqMode ,_
                BackOff ,_
                StartPower ,_
                StopPower ,_
                NumPowers ,_
                CompAlg ,_
                DwellTime ,_
                IFBAndwidth ,_
                ShowIterations )

call CalGCA ( parser )
call Analysis( parser )

```

```
.....  
' ' GCA Setup  
.....
```

```
sub SetupGCA ( parser , StartFreq , StopFreq , NumFreqs ,  
EnableInterp , Scale , CompLevel , LinearPower ,_  
  
            AcqMode , BackOff , StartPower , StopPower , NumPowers  
, CompAlg , DwellTime , IFBAndwidth ,_  
  
            ShowIterations )  
  
parser . Parse "*RST "  
  
    parser . Parse "CALC:PAR:DEL:ALL "  
  
    parser . Parse "CALC:CUST:DEF " "S21" " , "Gain Compression" " , "S21" "  
"  
  
    parser . Parse "DISP:WIND:TRAC1:FEED " "S21" " "  
  
        parser . Parse "CALC:PAR:SEL " "S21" " "  
  
            parser . Parse "CALC:CUST:DEF " "CompIn21" " , "Gain  
Compression" " , "CompIn21" " "  
  
                parser . Parse "DISP:WIND:TRAC2:FEED " "CompIn21" " "  
  
                    parser . Parse "CALC:CUST:DEF " "DeltaGain21" " , "Gain  
Compression" " , "DeltaGain21" " "  
  
                        parser . Parse "DISP:WIND:TRAC3:FEED " "DeltaGain21" " "  
  
                            parser . Parse "SENS:SWE:MODE HOLD"  
  
                                parser . Parse "DISP:WIND1:TRAC3:Y:SCAL:PDIV " & Scale  
  
                                    parser . Parse "DISP:WIND1:TRAC3:Y:RLEV " & -CompLevel  
  
select case AcqMode  
  
case 0 ' SMART Sweep  
  
    parser . Parse "SENS:GCS:AMOD SMAR"  
  
case 1 ' 2D Power Sweeps  
  
    parser . Parse "SENS:GCS:AMOD PFREQ"
```

```
case 2 ' 2D Freq Sweeps
  parser . Parse "SENS:GCS:AMOD FPOW"
end select
select case CompAlg
case 0 ' Deviation from linear gain
  parser . Parse "SENS:GCS:COMP:ALG CFLG"
case 1 ' Deviation from max gain
  parser . Parse "SENS:GCS:COMP:ALG CFMG"
case 2 ' Back Off
  parser . Parse "SENS:GCS:COMP:ALG BACK"
case 3 ' XY
  parser . Parse "SENS:GCS:COMP:ALG XYCOM"
end select

if EnableInterp then
  parser . Parse "SENS:GCS:COMP:INT ON"
else
  parser . Parse "SENS:GCS:COMP:INT OFF"
end if

if ShowIterations then
  parser . Parse "SENS:GCS:SMAR:SIT ON"
else
  parser . Parse "SENS:GCS:SMAR:SIT OFF"
end if
```

```
parser . Parse "SENS:GCS:COMP:LEV " & CompLevel
parser . Parse "SENS:GCS:COMP:BACK:LEV " & BackOff
parser . Parse "SENS:GCS:COMP:DELT:X " & BackOff
parser . Parse "SENS:GCS:COMP:DELT:Y " & BackOff
parser . Parse "SENS:GCS:SWE:FREQ:POIN " & NumPowers
parser . Parse "SENS:GCS:SMAR:STIM " & DwellTime
parser . Parse "SENS:BAND " & IFBandwidth
parser . Parse "SENS:SWE:DWEL " & DwellTime
parser . Parse "SOUR:POW:STAR " & StartPower
parser . Parse "SOUR:POW:STOP " & StopPower
parser . Parse "SENS:FREQ:STAR " & StartFreq
parser . Parse "SENS:FREQ:STOP " & StopFreq
parser . Parse "SENS:SWE:POIN " & NumFreqs
parser . Parse "SENS:SWE:MODE SING"
```

```
dim str
```

```
str = parser .Parse ( "* OPC ?" )
```

```
end sub
```

```
.....
'' GCA Calibration
.....
```

```
sub CalGCA ( parser )
```

```
dim CalSteps , I
```

```
parser . parse "SENS:CORR:COLL:GUID:CONN:PORT1 'APC 3.5 female'"
```

```

parser . parse "SENS:CORR:COLL:GUID:CONN:PORT2 'APC 3.5 male'"
parser . parse "SENS:CORR:COLL:GUID:CKIT:PORT1 'N4691-60004 ECal
'"
parser . parse "SENS:CORR:COLL:GUID:CKIT:PORT2 'N4691-60004 ECal
'"
parser . parse "SENS:CORR:GCSetup:POW 0"
parser . parse "SENS:CORR:COLL:GUID:INIT "
CalSteps = parser . parse ( " SENS:CORR:COLL:GUID:STEP ?" )
for I = 1 to CalSteps
msgBox parser .parse ( "SENS:CORR:COLL:GUID:DESC ? " & I )
parser . parse ( "SENS:CORR:COLL:GUID:ACQ STAN"& I )
next
parser . parse "SENS:CORR:COLL:GUID:SAVE "
msgBox "Done"
end sub

```

```

.....
'' GCA Analysis
.....

```

```

sub Analysis( parser )
'select measurement 1
parser.parse "CALC:PAR:MNUM 1"
parser.parse "CALC:GCM:ANAL:ENABLE 1" 'turn on the analysis mode
parser.parse "CALC:GCM:ANAL:CWFR 1e9" 'set the analysis cw frequency
'select measurement 2
parser.parse "CALC:PAR:MNUM 2"
parser.parse "CALC:GCM:ANAL:ENABLE 1"

```

```
parser.parse "CALC:GCM:ANAL:CWFR 2e9"  
parser.parse "CALC:GCM:ANAL:XAX PSO" 'set the axis to power settings  
'select measurement 3  
parser.parse "CALC:PAR:MNUM 3"  
parser.parse "CALC:GCM:ANAL:ENABLE 1"  
parser.parse "CALC:GCM:ANAL:CWFR 3e9"  
parser.parse "CALC:GCM:ANAL:ISD 0" ' set the discrete frequency  
option to false  
e nd sub
```

Last Modified:

3-Sep-2009	Added Analysis
1712-Mar-2014	MX New topic

Create and Cal a GCX Measurement

This VBS program does the following:

- creates and configures GCX
- performs a calibration using an ECal with 3.5 mm Female on Port A and 3.5 mm Male connectors on Port B

To run this example without modification you need the following:

- An ECal module that covers the frequency range of the measurement. You may need to change the ECal model number in the program.
- A power meter must be available to the PNA. This can be accomplished either by attaching the meter to the PNA via a GPIB cable, or by using SCPI over LAN.

By removing the comments (') at the start of the **BLUE** code, it can also do the following:

- Use ECal characterizations
- Specify Mechanical Cal Kits
- Perform manual ECal orientation.
- Specify the thru measurement method.
- Omit the isolation part of the 2-port cal.
- Perform an LO Power Cal.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example. However, some modification is necessary to make the program run on a traditional GPIB Interface. For example, during the power meter portion of this calibration, scpi.Parse will not process a command until the power meter routine has completed. Traditional GPIB would require a serial polling technique to ensure the routine has completed before proceeding.

This program can be run as a macro in the PNA. To do this, copy the code into a text editor file such as Notepad and save on the PNA hard drive as GCX.vbs. [Learn how to setup and run the macro.](#)

See the [Gain Compression](#) and [Mixer](#) Commands

See other [SCPI example programs](#)

```

option explicit

dim CompLevel, Tolerance, StartFreq, StopFreq, LOFreq, NumFreqs, Scale,
LinearPower

dim AcqMode, BackOff, StartPower, StopPower, NumPowers, EnableInterp, CompAlg

dim DwellTime, IFBandwidth, ShowIterations, host, app, parser, i

CompLevel      = 1      ' 1 dB compression level
Tolerance      = 0.05   ' SMART Sweep tolerance
NumFreqs       = 21
Scale          = 0.1
LinearPower    = -10
BackOff        = 10     ' Not used for Deviation from linear gain
StartPower     = -20
StopPower      = 8
NumPowers      = 60     ' Not used for SMART Sweep
DwellTime      = 0.0005 ' Allow some time for DUT bias/thermal effects
IFBandwidth    = 1000   ' Reasonable trace noise at -20 dBm
EnableInterp   = False  ' Disable interpolation
AcqMode        = 0      ' Smart Sweep
CompAlg        = 0      ' Deviation from linear gain
ShowIterations = False  ' Configure SMART to not show iteration results

dim objargs

set objargs = wscript.Arguments

if (objArgs.Count = 1) then host = objargs(0)

.....

'' Create and Configuration GCX Channel:
.....

set app = CreateObject("Agilentpna835x.application")

set parser = app.ScpiStringParser

```

```
.....  
' ' GCX Setup  
.....
```

```
parser.Parse "*RST"  
  
parser.Parse "CALC:PAR:DEL:ALL"  
  
parser.Parse "CALC:CUST:DEF ""SC21"", "Gain Compression Converters", "SC21" "  
  
parser.Parse "DISP:WIND:TRAC1:FEED ""SC21"" "  
  
parser.Parse "CALC:PAR:SEL ""SC21"" "  
  
parser.Parse "CALC:CUST:DEF ""CompIn21"", "Gain Compression  
Converters", "CompIn21" "  
  
parser.Parse "DISP:WIND:TRAC2:FEED ""CompIn21"" "  
  
parser.Parse "CALC:CUST:DEF ""DeltaGain21"", "Gain Compression  
Converters", "DeltaGain21" "  
  
parser.Parse "DISP:WIND:TRAC3:FEED ""DeltaGain21"" "  
  
parser.Parse "SENS:SWE:MODE HOLD"  
  
parser.Parse "DISP:WIND1:TRAC3:Y:SCAL:PDIV " & Scale  
parser.Parse "DISP:WIND1:TRAC3:Y:RLEV " & -CompLevel  
  
select case AcqMode  
  
case 0 ' SMART Sweep  
  
parser.Parse "SENS:GCS:AMOD SMAR"  
  
case 1 ' 2D Power Sweeps  
  
parser.Parse "SENS:GCS:AMOD PFREQ"  
  
case 2 ' 2D Freq Sweeps  
  
parser.Parse "SENS:GCS:AMOD FPOW"  
  
end select  
  
select case CompAlg  
  
case 0 ' Deviation from linear gain  
  
parser.Parse "SENS:GCS:COMP:ALG CFLG"  
  
case 1 ' Deviation from max gain
```

```
parser.Parse "SENS:GCS:COMP:ALG CFMG"

case 2 ' Back Off

parser.Parse "SENS:GCS:COMP:ALG BACK"

case 3 ' XY

parser.Parse "SENS:GCS:COMP:ALG XYCOM"

end select

if EnableInterp then

parser.Parse "SENS:GCS:COMP:INT ON"

else

parser.Parse "SENS:GCS:COMP:INT OFF"

end if

if ShowIterations then

parser.Parse "SENS:GCS:SMAR:SIT ON"

else

parser.Parse "SENS:GCS:SMAR:SIT OFF"

end if

parser.Parse "SENS:GCS:COMP:LEV " & CompLevel

parser.Parse "SENS:GCS:COMP:BACK:LEV " & BackOff

parser.Parse "SENS:GCS:COMP:DELT:X " & BackOff

parser.Parse "SENS:GCS:COMP:DELT:Y " & BackOff

parser.Parse "SENS:GCS:SWE:FREQ:POIN " & NumPowers

parser.Parse "SENS:GCS:SMAR:STIM " & DwellTime

parser.Parse "SENS:BAND " & IFBandwidth

parser.Parse "SENS:SWE:DWEL " & DwellTime

parser.Parse "SOUR:POW:STAR " & StartPower
```

```

parser.Parse "SOUR:POW:STOP " & StopPower

parser.Parse "SOUR:POW " & LinearPower

parser.Parse "SENS:SWE:POIN " & NumFreqs

' Mixer settings

parser.Parse "SENS:MIX:INPut:FREQ:MODE SWEPT"

parser.Parse "SENS:MIX:INPut:FREQ:STAR 3.6e9"

parser.Parse "SENS:MIX:INPut:FREQ:STOP 3.9e9"

parser.Parse "SENS:MIX:LO:FREQ:MODE FIXED"

parser.Parse "SENS:MIX:LO:FREQ:FIX 1e9"

parser.Parse "SENS:MIX:LO:POW 10"

parser.Parse "SENS:MIX:OUTP:FREQ:SID LOW"

parser.Parse "SENS:MIX:CALC Output"

parser.Parse "SENS:MIX:APPLY"

'First apply the settings, then set LO Name

parser.Parse "SENS:MIX:LO:NAME 'Port 3'"

parser.Parse "SENS:MIX:APPLY"

parser.Parse "SENS:MIX:SAVE "C:/Program Files/Agilent/Network
Analyzer/Documents/Mixer/MyMixer.mxr"

' sweep single

parser.Parse "SENS:SWE:MODE SING"

dim str

str =parser.Parse("*OPC?")

.....

'' GCX Calibration

```

```

.....

'-----Perform A GCX Cal using SMC commands-----

'Specify the connector types and the cal kits for each of the ports.

parser.Parse "SENS:CORR:COLL:GUID:CONN:PORT1:SEL ""APC 3.5 male""

parser.Parse "SENS:CORR:COLL:GUID:CONN:PORT2:SEL ""APC 3.5 female""

parser.Parse "SENS:CORR:COLL:GUID:CKIT:PORT1:SEL ""N4691-60004 ECal""

parser.Parse "SENS:CORR:COLL:GUID:CKIT:PORT2:SEL ""N4691-60004 ECal""

' Non-factory characterizations are specified as follows:

'parser.Parse "sens:corr:coll:guid:ckit:port2 'N4691-60004 User 1 ECal'"

' When two or more ECal modules with the same model number are connected

' also specify the serial number as follows:

'parser.Parse "sens:corr:coll:guid:ckit:port2 'N4691-60004 ECal 01234'"

' When Disk Memory ECal user characterizations are used,

' specify both the User char and the serial number as follows:

'parser.Parse "sens:corr:coll:guid:ckit:port2 'N4691-60004 MyDskChar ECal 01234'"

' Uncomment the following lines to manually orient

' the ecal port A connected to PNA port 1

'parser.Parse "SENS:CORR:PREF:ECAL:ORI OFF"

'parser.Parse "SENS:CORR:PREF:ECAL:PMAP ECAL2 ="A1,B2"

' Specify Mechanical cal kits

'parser.Parse "sens:corr:coll:guid:ckit:port1 '85033D/E'"

'parser.Parse "sens:corr:coll:guid:ckit:port2 '85033D/E'"

'Optional settings

'Specify the thru measurement method.

'Always send the INIT command before this command.

```

```

'parser.Parse "SENS:CORR:COLL:GUID:INIT"

'parser.Parse "SENS:CORR:COLL:GUID:PATH:TMET 1,2,""UNDEFINED THRU""

'Omit the isolation part of the 2-port cal (default behavior).

'parser.Parse "SENS:CORR:COLL:GUID:ISOL NONE"

'Perform LO Power Cal

'parser.Parse "SENS:CORR:COLL:GUID:SMC:LO1:PCAL 1"

'Initialize a Guided calibration.

parser.Parse "SENS:CORR:COLL:GUID:INIT"

'Tell the wizard to generate and report the number of steps in this cal.

Dim steps

Dim desc

'Determine the number of steps required to complete the calibration.

steps =parser.Parse ("SENS:CORR:COLL:GUID:STEP?")

For i = 1 To steps

'Display the prompt for each step

desc =parser.Parse ("SENS:CORR:COLL:GUID:DESC? " & CStr(i))

MsgBox (desc)

'Perform the measurement for each step

parser.Parse "SENS:CORR:COLL:GUID:ACQ STAN" & CStr(i)

Next

'Finish the cal and save the calset

parser.Parse ("SENS:CORR:COLL:GUID:SAVE ON")

Msgbox ("GCX cal saved to CH1_CALREG")

```

Last Modified:

10-Jan-2013 Added INIT before tmet
6-Oct-2011 Updated with new SMC commands
19-Oct-2010 MX New topic

Create and Cal a Noise Figure Measurement

This example program creates a Noise Figure measurement, then calibrates the measurement.

You MUST change the ECal Identification strings (in **Blue** font).

Optional: Uncomment the following lines (in **Blue** font) to change these settings:

- Noise Receiver = Noise Receiver to Std (PNA) Receiver
- Cal Method = "Vector" to "Scalar"
- Receiver Characterization Method = "NoiseSource" to "PowerMeter"

This VBScript program can be run as a macro in the PNA. To do this, copy the code into a text editor file such as Notepad and save on the PNA hard drive as NF.vbs. [Learn how to setup and run the macro.](#)

[See the Noise figure commands.](#)

See Other SCPI Example Programs

```
' This section gets the PNA application
' starts the scpi parser, and presets the PNA
windowNum = 1
channelNum = 1
set pna=CreateObject("AgilentPNA835x.Application")
set scpi = pna.ScpiStringParser
' Create noise figure measurement
scpi.Parse "SYST:FPR"
scpi.Parse "DISP:WIND ON"
scpi.Parse "CALC:CUST:DEF 'noiseFig', 'Noise Figure Cold Source',
'NF'"
scpi.Parse "DISP:WIND:TRAC:FEED 'noiseFig'"
scpi.Parse "CALC:PAR:SEL 'noiseFig'"
```

```

' Substitute appropriate Ecal identification strings here
tunerEcal = "N4691-60004 ECal 02821"
pullEcal = "N4691-60004 ECal 02297"
' configure channel
ConfigureChannel
ConfigureNoiseSettings
' perform calibration
SetupNoiseSource
SetupCalAttributes_Insertable
FinishCalibration
' ----- Support subroutines -----
' Configure noise channel
sub ConfigureChannel
    scpi.Parse "SENS:FREQ:START 750MHz"
    scpi.Parse "SENS:FREQ:STOP 5.0GHz"
    scpi.Parse "SENS:SWEEP:POINTS 401"
    scpi.Parse "SENS:BWID 1.0E3"
end sub
' Configure noise-specific channel settings
sub ConfigureNoiseSettings
    scpi.Parse "SENS:NOIS:REC NOISe"      'Use noise receivers
' scpi.Parse "SENS:NOIS:REC NORM"      'Use std PNA receiver
    scpi.Parse "SENS:NOIS:AVER:STAT ON"   ' turn averaging ON
    scpi.Parse "SENS:NOIS:AVER 40"       ' noise averaging

    scpi.Parse "SENS:NOIS:BWID 8MHz"     ' noise bandwidth
    scpi.Parse "SENS:NOIS:GAIN 30"      ' gain of noise

```

receiver

scpi.Parse "SENS:NOIS:TEMP:AMB 301" ' ambient temperature, in Kelvin

scpi.Parse "SENS:NOIS:IMP:COUN 5" ' number of tuner impedance states

scpi.Parse "SENS:NOIS:TUN:ID '" & tunerEcal & "'" ' set ID of tuner Ecal module

scpi.Parse "SENS:NOIS:TUN:INP 'B'" ' orientation of tuner input port

scpi.Parse "SENS:NOIS:TUN:OUTP 'A'" ' orientation of tuner output port

scpi.Parse "SENS:CORR:TCOL:USER:VAL 300" ' noise source cold temperature

end sub

sub SetupCalAttributes_Insertable

scpi.Parse "SENS:CORR:COLL:GUID:CONN:PORT1 'APC 3.5 female'"

scpi.Parse "SENS:CORR:COLL:GUID:CONN:PORT2 'APC 3.5 male'"

scpi.Parse "SENS:CORR:COLL:GUID:CKIT:PORT1 '" & pullEcal & "'" ' port 1 calkit

scpi.Parse "SENS:CORR:COLL:GUID:CKIT:PORT2 '" & pullEcal & "'" ' port 2 calkit

scpi.Parse "SENS:NOIS:SOUR:CONN 'APC 3.5 male'" ' noise source connector type

scpi.Parse "SENS:NOIS:SOUR:CKIT '" & pullEcal & "'" ' noise source calkit

scpi.Parse "SENS:NOISE:CAL:METHOD 'Vector'" ' cal method

' scpi.Parse "SENS:NOISE:CAL:METHOD 'Scalar'"

scpi.Parse "SENS:NOISE:CAL:RMETHOD 'NoiseSource'" 'Receiver

Characterization method

```
' scpi.Parse "SENS:NOISE:CAL:RMETHOD 'PowerMeter'"
  scpi.Parse "SENS:CORR:COLL:GUID:INIT"
end sub

sub SetupNoiseSource
  ' specify the ENR file for the noise source
  enrfile = "C:/Program Files/Agilent/Network
Analyzer/Noise/346C_MY44420454.enr"
  scpi.Parse "SENS:NOIS:ENR:FILENAME '" & enrfile & "'"
  ' set noise source cold temperature
  scpi.Parse "SENS:CORR:TCOLD:USER:VAL 301.1"
end sub

sub FinishCalibration
  ' Build the connection list and acquire the calibration
  steps = scpi.Parse("SENS:CORR:COLL:GUID:STEPS?")
  for i = 1 to steps
    str = scpi.Parse("SENS:CORR:COLL:GUID:DESC? " & i)
    msgbox str
    scpi.Parse "SENS:CORR:COLL:GUID:ACQ STAN" & i
  next
  scpi.Parse "SENS:CORR:COLL:GUID:SAVE 0"
  wscript.echo "Calibration complete"
end sub
```

Last Modified:

9-Jun-2011 Updated with Rcvr Char Method A.09.41.

Create and Cal a VMC Measurement

The following example program sets up a 1-stage mixer, then performs a full VMC calibration.

By removing the comments (') at the start of the **BLUE code**, it can also do the following:

- Load a mixer setup file
- Use an ECal Module
- Perform manual ECal orientation
- Load a Mixer Characterization

See Also

[Setup Converter commands](#)

[VMC Cal commands](#)

[All Guided Cal commands](#)

Example - [Perform a Mixer Characterization ONLY](#)

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as VMC.vbs. [Learn how to setup and run the macro.](#)

```
Dim app

Dim scpi

' Create / Get the PNA application.

Set app = CreateObject("AgilentPNA835x.Application")

Set scpi = app.ScpiStringParser

scpi.Parse "SYSTEM:PRESet"

'Create a Vector Mixer Measurement

'First, delete all measurements on the channel

scpi.Parse "CALC:PAR:DEL:ALL"
```

'Create a forward scalar mixer measurement and configure it in channel 1.

'The first parameter is a unique identifying string to allow subsequent

'commands to be directed at this specific measurement.

```
scpi.Parse "CALC:CUST:DEF 'My VC21', 'Vector Mixer/Converter', 'VC21'"
```

'Setup the new measurement as the 2nd trace in the active window

```
scpi.Parse "DISP:WIND:TRAC2:FEED 'My VC21'"
```

'Make the new trace the active measurement

```
scpi.Parse "CALC:PAR:SEL 'My VC21'"
```

'The parameters of the mixer measurement can now be configured.

'This can be done by either using the SENS:MIX commands

'for each of the parameters or by loading a mixer setup file.

'Uncomment the following line to load a mixer setup file. The path

'name for the mixer file may be loaded from other mapped drives.

```
'scpi.Parse "SENS:MIXer:Load 'c:\users\public\network  
analyzer\documents\Mixer/MyMixer.mxr'"
```

' Setup Stimulus

' Points and IFBW are channel settings

```
scpi.Parse "SENS:SWEep:POINTs 21"
```

```
scpi.Parse "SENS:BANDwidth 1e3"
```

' The rest are mixer settings

```
scpi.Parse "SENS:MIX:LO:FREQ:MODE SWEpt"
```

```
scpi.Parse "SENS:MIX:INPut:FREQ:STAR 3.6e9"
```

```
scpi.Parse "SENS:MIX:INPut:FREQ:STOP 3.9e9"
```

```
scpi.Parse "SENS:MIX:LO:FREQ:MODE FIXED"
```

```
scpi.Parse "SENS:MIX:LO:FREQ:FIX 1e9"
```

```
scpi.Parse "SENS:MIX:LO:POW 10"
```

```
scpi.Parse "SENS:MIX:OUTP:FREQ:SID LOW"
```

```
scpi.Parse "SENS:MIX:CALC Output"

scpi.Parse "SENS:MIX:LO:NAME 'Port 3'"

scpi.Parse "SENS:MIX:APPLY"

' Perform Cal

' Define the DUT connectors for at ports 1 and 2 of the PNA
scpi.Parse "sens:corr:coll:guid:conn:port1 'APC 3.5 female'"
scpi.Parse "sens:corr:coll:guid:conn:port2 'APC 3.5 male'"
scpi.Parse "sens:corr:coll:guid:conn:port3 'Not used'"
scpi.Parse "sens:corr:coll:guid:conn:port4 'Not used'"

' Specify Mechanical cal kits
scpi.Parse "sens:corr:coll:guid:ckit:port1 '85033D/E'"
scpi.Parse "sens:corr:coll:guid:ckit:port2 '85033D/E'"

' Specify an ECal module the same way
'scpi.Parse "sens:corr:coll:guid:ckit:port1 'N4691-60004 ECal'"
'scpi.Parse "sens:corr:coll:guid:ckit:port2 'N4691-60004 ECal'"

' Non-factory characterizations are specified as follows:
'scpi.Parse "sens:corr:coll:guid:ckit:port2 'N4691-60004 User 1 ECal'"

' When two or more ECal modules with the same model number are connected
' also specify the serial number as follows:
'scpi.Parse "sens:corr:coll:guid:ckit:port2 'N4691-60004 ECal 01234'"

' When Disk Memory ECal user characterizations are used,
' specify both the User char and the serial number as follows:
'scpi.Parse "sens:corr:coll:guid:ckit:port2 'N4691-60004 MyDskChar ECal 01234'"

'
```

```

' By default, VMC requires the measurement of a Calibration Mixer.
' To determine the conversion loss of the calmixer, the cal wizard
' will add a step to perform a 1 port cal at the output of the mixer.
' The following commands opt to perform the mixer
' characterization using a cal kit.

scpi.Parse "SENS:CORR:COLL:GUID:VMC:MIX:CHAR:CAL:OPT CKIT"

' Define the DUT connectors for the output of the characterization mixer
' Use (logical) Port 3. If it is already used by the DUT,
' then specify port 4.

scpi.Parse "sens:corr:coll:guid:conn:port3 'APC 3.5 male'"

' Specify the mechanical cal kit for port 3

scpi.Parse "sens:corr:coll:guid:ckit:port3 '85033D/E'"

' To avoid performing the 1-port cal, provide the cal wizard with a
' mixer characterization file. Uncomment the following line to
' specify the characterization file. This S2P file will be read.

'scpi.Parse "SENS:CORR:COLL:GUID:VMC:MIX:CHAR:CAL:OPT
FILE,'c:\users\public\network analyzer\documents\MyMixer.s2p'"

' ECal orientation

' By default, auto orientation of the ecal module is performed

' Uncomment the following lines to manually orient the ecal

'scpi.Parse "SENS:CORR:PREF:ECAL:ORI OFF"

' for 2-port portion, ecal port A connected to PNA port 1

'scpi.Parse "SENS:CORR:PREF:ECAL:PMAP ECAL2 ="A1,B2"

'for mixer char, ecal port A connected to cal mixer output

'scpi.Parse "SENS:CORR:COLL:GUID:VMC:MIXer:ECAL:PORTmap 1,"A1"

' the main calibration loop

```

```
' a description for the connection instructions is read
' and then the standard is acquired

dim steps, strPrompt
scpi.Parse "sens:corr:coll:guid:init"
steps=scpi.Parse ("sens:corr:coll:guid:steps?")
wscript.echo "Number of Steps = " + cstr(steps)
if (steps > 0) then ' otherwise an error condition occurred
for i = 1 to steps
    strPrompt = scpi.Parse ("sens:corr:coll:guid:desc? " + CStr(i))
    MsgBox strPrompt, vbOKOnly, step
    scpi.Parse ("sens:corr:coll:guid:acq STAN" + CStr(i))
next
scpi.Parse "sens:corr:coll:guid:save"
MsgBox ("Cal is done!")

end if
```

Last Modified:

19-Feb-2013	Remove Mixer Char only
1-Dec-2011	Added commands to choose cal kit for char mixer
12-Sep-2011	Fix port 2 Mech Cal kit
5-Apr-2011	Edited for ECal
16-Feb-2011	Modifications for A.09.33

Create and Cal a Swept IMD Measurement

This program does the following:

- Create IMD power and IM3 measurements
- Set sweep mode to Center Frequency Sweep
- Calibrate the IMD channel

This program can be run as a macro in the PNA. To do this, copy the code into a text editor file such as Notepad and save on the PNA hard drive as IMD.vbs.

See Also

[Learn how to setup and run the macro.](#)

[See the IMD commands.](#)

[See the IMD Cal commands](#)

See Other SCPI Example Programs

```
option explicit
'declare variables
dim SweepMode, StartDeltaFreq, StopDeltaFreq, NumFreqs, TonePower, CWFreq
dim app, hostname, parser
'' Sweep type:
'' naIMDToneCWSweep          = 0
'' naIMDTonePowerSweep      = 1
'' naIMDToneCenterFreqSweep = 2
'' naIMDDeltaFrequencySweep = 3
'' naIMDToneSegmentSweep   = 4
'init variables
SweepMode    = 3          ' Sweep DeltaF
StartDeltaFreq = 100e3
```

```

StopDeltaFreq    = 1e9

NumFreqs        = 201

TonePower       = -7

CWFreq          = 5e9

' get host name from commandline

dim objargs

set objargs = wscript.arguments

if(objargs.Count = 1) then hostname = objargs(0)

set app = CreateObject("Agilentpna835x.application", hostname)

set parser = app.ScpiStringParser

call SetupIMD

call CalIMD

.....

'' Create and Configure IMD channel

.....

sub SetupIMD

    parser.Parse "*RST"

    parser.Parse "CALC:PAR:DEL:ALL"

    parser.Parse "CALC:CUST:DEF 'PwrMain','Swept IMD', 'PwrMain'" ' create
PwrMain measurement

    parser.Parse "DISP:WIND:TRAC1:FEED 'PwrMain'"

    parser.Parse "CALC:PAR:SEL 'PwrMain'"

    parser.Parse "CALC:CUST:DEF 'IM3', 'Swept IMD', 'IM3'" ' create IM3
measurement

    parser.Parse "DISP:WIND:TRAC2:FEED 'IM3'"

    parser.Parse "SENS:SWE:MODE HOLD"

' set sweep mode

select case SweepMode

```

```

case 0 ' CW sweep

    parser.Parse "SENS:IMD:SWE:TYPE CW"

case 1 ' Power Sweep

    parser.Parse "SENS:IMD:SWE:TYPE POW"

case 2 ' sweep Fc

    parser.Parse "SENS:IMD:SWE:TYPE FCEN"

case 3 ' sweep DeltaF

    parser.Parse "SENS:IMD:SWE:TYPE DFR"

case 4 ' segment sweep

    parser.Parse "SENS:IMD:SWE:TYPE SEGM"

end select

parser.Parse "SENS:IMD:FREQ:FCEN " & CWFreq           ' Frequency Center

parser.Parse "SENS:IMD:FREQ:DFR:STAR " & startDeltaFreq ' Delta
Frequency Start

parser.Parse "SENS:IMD:FREQ:DFR:STOP " & stopDeltaFreq ' Delta
Frequency Stop

parser.Parse "SENS:IMD:TPOW:F1 " & TonePower           ' F1 power

parser.Parse "SENS:IMD:TPOW:F2 " & TonePower           ' F2 power

parser.Parse "SENS:SWE:POIN " & NumFreqs

parser.Parse "SENS:SWE:MODE SING"

end sub

sub CalIMD

    'Configure IMD GuidedCal for the connector types and ECal module that will be
    used

    ' Substitute appropriate connector type and ECal identification strings here

```

```
parser.Parse "SENS:CORR:COLL:GUID:CONN:PORT1 'APC 3.5 female'"
```

```
parser.Parse "SENS:CORR:COLL:GUID:CONN:PORT2 'APC 3.5 male'"
```

```
parser.Parse "SENS:CORR:COLL:GUID:CKIT:PORT1 'N4693-60001 User 2 ECal 00012'"
```

```
parser.Parse "SENS:CORR:COLL:GUID:CKIT:PORT2 'N4693-60001 User 2 ECal 00012'"
```

```
' imd custom settings
```

```
' Set the Power Level at the power sensor to be used in calibration
```

```
parser.Parse "SENS:CORR:IMD:POW 0"
```

```
' Specify the connector type of the power sensor. If there is an adapter  
between
```

```
' the input port and the power sensor, specify the connector type here, and  
set
```

```
' the appropriate cal kit type for the connector so that extra calibration  
can be
```

```
' performed. To skip the calibration for the adapter, set the connection  
type to "Ignored"
```

```
' i.e.: SENS:CORR:IMD:SENS:CONN 'Ignored'
```

```
parser.Parse "SENS:CORR:IMD:SENS:CONN 'APC 3.5 female'"      'set power sensor  
connector type
```

```
parser.Parse "SENS:CORR:IMD:SENS:CKIT 'N4693-60001 User 2 ECal 00012'"      '  
set power sensor calkit type
```

```
' Set the Max product to calibrate, valid values are 3, 5, 7, and 9
```

```
parser.Parse "SENS:CORR:IMD:MPR 3"
```

```
' Set the calibration Frequencies, can choose between calibrate only at  
center Frequencies (CENT)
```

```
' or calibrate at all frequencies (ALL).
```

```
parser.Parse "SENS:CORR:IMD:CAL:FREQ ALL"
```

```
'Include 2nd order product in calibration

parser.Parse "SENS:CORR:IMD:SORD:INCL 1"

parser.Parse "SENS:CORR:COLL:GUID:INIT"

dim CalSteps, I

CalSteps = parser.Parse("SENS:CORR:COLL:GUID:STEP?")

for I = 1 to CalSteps

    msgBox parser.Parse("SENS:CORR:COLL:GUID:DESC? " & I)

    parser.Parse("SENS:CORR:COLL:GUID:ACQ STAN" & I)

next

parser.Parse "SENS:CORR:COLL:GUID:SAVE"

msgBox "IMD Cal Done"

end sub
```

Create and Cal an NFX Measurement

This program does the following:

- Setup a Noise Figure SC21 Measurement
- Calibrate Noise Figure channel
- Optional - Configure for an Embedded LO

To run this program, make the following edits, highlighted in **yellow**:

- Set **hostname** to your PNA computer name
- Set **tunerECal** and **pullECal** to your ECal model and info
- Set **ENR** to correct file name and location
- Set **connector types** for ECal, power sensor, and noise source

This program can be run as a macro in the PNA. To do this, copy the code into a text editor file such as Notepad and save on the PNA hard drive as NFX.vbs. [Learn how to setup and run the macro.](#)

See SCPI commands

[Calc:Custom command](#)

[Noise Figure commands](#)

[Mixer commands](#)

[Embedded LO commands](#)

[Guided Cal commands](#)

[Noise Cal commands](#)

[See Other SCPI Example Programs](#)

```
option explicit
dim app
dim hostname
```

```
dim parser

hostname = "MyPNA"

set app = CreateObject("Agilentpna835x.application", hostname)
set parser = app.ScpStringParser

' ecal and noise tuner

dim tunerEcal

dim sParamEcal

tunerEcal = "N4691-60003 ECal 00591"

sParamEcal = "N4693-60001 User 2 ECal 00012"

' ENR file

dim ENRFile

ENRFile = "C:/Program Files/Agilent/Network
Analyzer/Noise/346C_44420601.enr"

call SetupNFX

'optional if not doing embedded LO

'call SetupEmbeddedLO

call CalNFX

sub SetupNFX

'Create NF and SC21 traces

parser.Parse "*RST"

parser.Parse "CALC:PAR:DEL:ALL"

parser.Parse "CALC:CUST:DEF 'NF', 'Noise Figure Converters', 'NF'
"

parser.Parse "DISP:WIND:TRAC1:FEED 'NF'"

parser.Parse "CALC:CUST:DEF 'SC21', 'Noise Figure Converters',
'SC21' "
```

```
parser.Parse "DISP:WIND:TRAC2:FEED 'SC21' "  
parser.Parse "SENS:SWE:MODE SING"  
  
'set channel properties  
' set sweep type to linear sweep  
parser.Parse "SENS:SWE:TYPE LIN"  
' set number of points  
parser.Parse "SENS:SWE:POIN 101"  
' set IFbandwidth  
parser.Parse "SENS:BWID 1e3"  
  
'set nfx properties  
' turn averaging on  
parser.Parse "SENS:NOIS:AVER:STAT ON"  
' noise averaging factor  
parser.Parse "SENS:NOIS:AVER 40"  
' noise tuner ecal module  
parser.Parse "SENS:NOIS:TUN:ID '' & tunerECal & '' "  
' noise tuner input  
parser.Parse "SENS:NOIS:TUN:INP 'B' "  
' noise tuner output  
parser.Parse "SENS:NOIS:TUN:OUTP 'A' "  
' noise bandwidth  
parser.Parse "SENS:NOIS:BWID 8e6"  
' low gain of noise receiver  
parser.Parse "SENS:NOIS:GAIN 0"
```

```
' sweep single

parser.Parse "SENS:SWE:MODE SING"

' Mixer settings

parser.Parse "SENS:MIX:INPut:FREQ:MODE SWEpt"
parser.Parse "SENS:MIX:INPut:FREQ:STAR 3.6e9"
parser.Parse "SENS:MIX:INPut:FREQ:STOP 3.9e9"
parser.Parse "SENS:MIX:LO:FREQ:MODE FIXED"
parser.Parse "SENS:MIX:LO:FREQ:FIX 1e9"
parser.Parse "SENS:MIX:LO:POW 10"
parser.Parse "SENS:MIX:OUTP:FREQ:SID LOW"
parser.Parse "SENS:MIX:CALC Output"
parser.Parse "SENS:MIX:APPLY"

'First apply the settings, then set LO Name

parser.Parse "SENS:MIX:LO:NAME 'Port 3'"
parser.Parse "SENS:MIX:APPLY"

parser.Parse "SENS:MIX:SAVE "C:/Program Files/Agilent/Network
Analyzer/Documents/Mixer/MyMixer.mrxr""

' sweep single

parser.Parse "SENS:SWE:MODE SING"

end sub

sub CalNFX

' dut connector

parser.Parse "SENS:CORR:COLL:GUID:CONN:PORT1 'APC 3.5 female'"
parser.Parse "SENS:CORR:COLL:GUID:CONN:PORT2 'APC 3.5 female'"

' port calkits
```

```
parser.Parse "SENS:CORR:COLL:GUID:CKIT:PORT1 '" & sParamECal & "'  
"  
parser.Parse "SENS:CORR:COLL:GUID:CKIT:PORT2 '" & sParamECal & "'  
"  
' power sensor connector  
parser.Parse "SENS:CORR:COLL:GUID:PSEN1:CONN 'APC 3.5 female'"  
' power sensor adapter calkit  
parser.Parse "SENS:CORR:COLL:GUID:PSEN1:CKIT '" & sParamECal & "'  
"  
' power calibration power level  
parser.Parse "SENS:CORR:COLL:GUID:PSEN1:POW:LEV -20"  
  
' noise source connector  
parser.Parse "SENS:NOIS:SOUR:CONN 'APC 3.5 male'"  
' noise source adapter cal calkit  
parser.Parse "SENS:NOIS:SOUR:CKIT '" & sParamECal & "' "  
' cal method  
parser.Parse "SENS:NOIS:CAL:METH 'Vector'"  
' ENR file  
parser.Parse "SENS:NOIS:ENR:FIL '" & ENRFile & "' "  
' disable LO power cal  
parser.Parse "SENS:CORR:COLL:NOIS:LO:PCAL 0"  
' set force both adapter calcs de-embed to false  
parser.Parse "SENS:CORR:COLL:NOIS:ENR:ADAP:DEEM 0"  
parser.Parse "SENS:CORR:COLL:NOIS:PSEN:ADAP:DEEM 0"  
' initialize guided cal
```

```
parser.Parse "SENS:CORR:COLL:GUID:INIT"

' step through calsteps
dim steps
steps = parser.Parse("SENS:CORR:COLL:GUID:STEP?")
dim i, str
for i = 1 to steps
str = parser.Parse("SENS:CORR:COLL:GUID:DESC? " & i)
msgbox str
parser.Parse "SENS:CORR:COLL:GUID:ACQ STAN" & i
next

parser.Parse "SENS:CORR:COLL:GUID:SAVE 0"
parser.Parse "SENS:SWE:MODE CONT"
end sub

sub SetupEmbeddedLO
' embedded LO properties
' normalize point
parser.Parse "SENS:MIX:ELO:NORM:POIN 101"
' set tuning mode to broadband and precise
parser.Parse "SENS:MIX:ELO:TUN:MODE BRO"
' tuning ifbw
parser.Parse "SENS:MIX:ELO:TUN:IFBW 3e4"
' max tuning iterations
parser.Parse "SENS:MIX:ELO:TUN:ITER 5"
```

```
' tuning tolerance
parser.Parse "SENS:MIX:ELO:TUN:TOL 1"
' tuning interval
parser.Parse "SENS:MIX:ELO:TUN:INT 1"
' turn on ELO
parser.Parse "SENS:MIX:ELO:STAT 1"
' sweep single
parser.Parse "SENS:SWE:MODE SING"
end sub
```

Last Modified:

29-Oct-2009 MX New topic

Create and Cal an SMC Measurement

This VB Script example creates and calibrates a scalar mixer measurement.

To run this example **without modification** you need the following:

- An ECal module that covers the frequency range of the measurement.
- A power meter must be available to the PNA. This can be accomplished either by attaching the meter to the PNA via a GPIB cable, or by using SCPI over LAN.

By removing the comments (') at the start of the **BLUE code**, it can also do the following:

- Load a mixer setup file
- Use ECal characterizations
- Specify Mechanical Cal Kits
- Perform manual ECal orientation.
- Specify the thru measurement method.
- Omit the isolation part of the 2-port cal.
- Perform an LO Power Cal.
- Enable and configure phase measurements

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example. However, some modification is necessary to make the program run on a traditional GPIB Interface. For example, during the power meter portion of this calibration, scpi.Parse will not process a command until the power meter routine has completed. Traditional GPIB would require a [serial polling technique](#) to ensure the routine has completed before proceeding.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as SMC.vbs. [Learn how to setup and run the macro.](#)

```
Dim app

Dim scpi

' Create / Get the PNA application.

Set app = CreateObject("AgilentPNA835x.Application")
```

```
Set scpi = app.ScpiStringParser

' Create a Scalar Mixer Forward Measurement
'First, delete all measurements on the channel
scpi.Parse "CALC:PAR:DEL:ALL"

'Create a forward scalar mixer measurement and configure it in
'channel 1. The first parameter is a unique
'identifying string (specified by the user) to allow subsequent
'commands to be directed at this specific measurement.
scpi.Parse "CALC:CUST:DEF 'My SC21', 'Scalar Mixer/Converter', 'SC21'"

'Setup the new measurement in the active window
scpi.Parse "DISP:WIND:TRAC:FEED 'My SC21'"

'Make the new trace the active measurement
scpi.Parse "CALC:PAR:SEL 'My SC21'"

'The parameters of the mixer measurement can now be configured.
'This can be done by either using the SENS:MIX commands
'for each of the parameters or by loading a mixer setup file.
'Uncomment the following line to load a mixer setup file. The path name
'for the mixer file may be loaded from other mapped drives.

'scpi.Parse "SENS:MIXer:Load 'c:\users\public\network
analyzer\documents\Mixer/MyMixer.mxr'"

' Setup Stimulus
' Points and IFBW are channel settings
scpi.Parse "SENS:SWEep:POINTs 21"
scpi.Parse "SENS:BANDwidth 1e3"

' Mixer settings
scpi.Parse "SENS:MIX:INPut:FREQ:MODE SWEPT"
```

```

scpi.Parse "SENS:MIX:INPut:FREQ:STAR 3.6e9"
scpi.Parse "SENS:MIX:INPut:FREQ:STOP 3.9e9"
scpi.Parse "SENS:MIX:LO:FREQ:MODE FIXED"
scpi.Parse "SENS:MIX:LO:FREQ:FIX 1e9"
scpi.Parse "SENS:MIX:LO:POW 10"
scpi.Parse "SENS:MIX:OUTP:FREQ:SID LOW"
scpi.Parse "SENS:MIX:CALC Output"
scpi.Parse "SENS:MIX:APPLY"

'First apply the settings, then set LO Name
scpi.Parse "SENS:MIX:LO:NAME 'Port 3'"
scpi.Parse "SENS:MIX:APPLY"

scpi.Parse "SENS:MIX:SAVE "C:/Program Files/Agilent/Network
Analyzer/Documents/Mixer/MyMixer.mxr"

'-----Perform A Scalar Mixer Calibration-----
'Specify the connector types and the cal kits for each of the ports.
scpi.Parse "SENS:CORR:COLL:GUID:CONN:PORT1:SEL ""APC 3.5 male""
scpi.Parse "SENS:CORR:COLL:GUID:CONN:PORT2:SEL ""APC 3.5 female""
scpi.Parse "SENS:CORR:COLL:GUID:CKIT:PORT1:SEL ""N4691-60004 ECal""
scpi.Parse "SENS:CORR:COLL:GUID:CKIT:PORT2:SEL ""N4691-60004 ECal""

' Non-factory characterizations are specified as follows:
'scpi.Parse "sens:corr:coll:guid:ckit:port2 'N4691-60004 User 1 ECal'"
' When two or more ECal modules with the same model number are connected
' also specify the serial number as follows:
'scpi.Parse "sens:corr:coll:guid:ckit:port2 'N4691-60004 ECal 01234'"
' When Disk Memory ECal user characterizations are used,
' specify both the User char and the serial number as follows:
'scpi.Parse "sens:corr:coll:guid:ckit:port2 'N4691-60004 MyDskChar ECal 01234'"

```

```
' Uncomment the following lines to manually orient
' the ecal port A connected to PNA port 1
'scpi.Parse "SENS:CORR:PREF:ECAL:ORI OFF"
'scpi.Parse "SENS:CORR:PREF:ECAL:PMAP ECAL2 ="A1,B2"

' Specify Mechanical cal kits
'scpi.Parse "sens:corr:coll:guid:ckit:port1 '85033D/E'"
'scpi.Parse "sens:corr:coll:guid:ckit:port2 '85033D/E'"

'Optional settings
'Specify the thru measurement method.
'Always send the INIT command before this command.
'scpi.Parse "SENS:CORR:COLL:GUID:INIT"
'scpi.Parse "SENS:CORR:COLL:GUID:PATH:TMET 1,2, ""UNDEFINED THRU"""
'Omit the isolation part of the 2-port cal (default behavior).
'scpi.Parse "SENS:CORR:COLL:GUID:ISOL NONE"

'Perform LO Power Cal
'scpi.Parse "SENS:CORR:COLL:GUID:SMC:LO1:PCAL 1"

'Enable and configure Phase measurements
'scpi.Parse "SENS:MIX:PHAS 1"
'scpi.Parse "SENS:MIX:NORM:POIN 1"

'Using Fixed delay
'scpi.Parse "SENS:CORR:COLL:GUID:SMC:PHAS:METH FIX"
'scpi.Parse "SENS:CORR:COLL:GUID:SMC:PHAS:DEL 12e-9"

'Initialize an SMC guided calibration.
scpi.Parse "SENS:CORR:COLL:GUID:INIT"
```

```

'Tell the wizard to generate and report the number of steps in this cal.

Dim steps

Dim desc

'Determine the number of steps required to complete the calibration.

steps = scpi.Parse ("SENS:CORR:COLL:GUID:STEP?")

For i = 1 To steps

'Display the prompt for each step

desc = scpi.Parse ("SENS:CORR:COLL:GUID:DESC? " & CStr(i))

MsgBox (desc)

'Perform the measurement for each step

scpi.Parse "SENS:CORR:COLL:GUID:ACQ STAN" & CStr(i)

Next

'Finish the cal and save the calset

scpi.Parse ("SENS:CORR:COLL:GUID:SAVE ON")

Msgbox ("SMC cal saved to CH1_CALREG")

```

Last Modified:

10-Jan-2013	Added init command before TMET
March 8, 2012	Updated with LO Name (ZW)
6-Oct-2011	Fixed mech cal kit typo, added phase and LO Pwr Cal
23-May-2011	Added mixer setup commands
5-Apr-2011	Edited ECal commands
8-Mar-2011	Updated with new SMC commands
4-Jun-2009	Added 'Done' command

Calibrate Multiple SMC Channels

This example allows you to calibrate multiple SMC channels while connecting the power meter and required standards or ECal module only once.

In the example program:

- Modify `chans = 2` to indicate the number of channels to calibrate.
- You can also change the connector type and cal kit for each port.

The SCPI commands in this example are sent over a COM interface using the `SCPIStringParser` object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as `MultChanCal.vbs`. [Learn how to setup and run the macro.](#)

```
Dim app
Dim scpi
Dim chans
Dim i
Dim steps
Dim desc

' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
app.Preset

' Set number of channels to create
chans = 2

ReDim calset(chans - 1)

For i = 1 To chans
    chanStr = CStr(i+1) ' calibrate on channels 2 and 3

    Dim parm, measName, sens, calc
```

```

parm = "S" & CStr(i) & CStr(i)

measName = "My" & parm

sens = "SENS" & chanStr

calc = "CALC" & chanStr

scpi.Parse calc & ":CUST:DEF '" & measName & "', 'Scalar Mixer/Converter', '"
& parm & "'"

'Setup the new measurement as the 2nd trace in the active window

scpi.Parse "DISP:WIND:TRAC" & chanStr & ":FEED '" & measName & "'"

'Make the new trace the active measurement

scpi.Parse calc & ":PAR:SEL '" & measName & "'"

'-----Perform A FCA Mixer Calibration-----

'Set ports and cal kits for 2 port calibration portion

scpi.Parse sens & ":CORR:COLL:GUID:CONN:PORT1:SEL ""APC 3.5 male""

scpi.Parse sens & ":CORR:COLL:GUID:CONN:PORT2:SEL ""APC 3.5 female""

scpi.Parse sens & ":CORR:COLL:GUID:CKIT:PORT1:SEL ""85052C""

scpi.Parse sens & ":CORR:COLL:GUID:CKIT:PORT2:SEL ""85052C""

'ECal modules are specified with the same command

' scpi.Parse sens & ":CORR:COLL:GUID:CKIT:PORT1:SEL ""N4691-60004 ECal""
' scpi.Parse sens & ":CORR:COLL:GUID:CKIT:PORT2:SEL ""N4691-60004 ECal""

'Specify the thru measurement method.

scpi.Parse sens & ":CORR:COLL:GUID:PATH:TMET 1,2, ""DEFINED THRU""

'Omit the isolation part of the 2-port cal

scpi.Parse sens & ":CORR:COLL:GUID:ISOL NONE"

```

```

'Initialize an SMC guided calibration.

scpi.Parse sens & ":CORR:COLL:GUID:INIT"

'Determine the number of steps required to complete the calibration.

steps = scpi.Parse (sens & ":CORR:COLL:GUID:STEP?")

Next

For j = 1 To CInt(steps)

'Display the prompt for each step

desc = scpi.Parse(sens & ":CORR:COLL:GUID:DESC? " & CStr(j))

MsgBox (desc)

'Measure the same standard for each channel

For i = 1 To chans

    chanStr = CStr(i+1) ' channel number as string

    scpi.Parse "SENS" & chanStr & ":CORR:COLL:GUID:ACQ STAN" & CStr(j)

    opc_comp = scpi.Parse("*OPC?")

Next

Next

'Finish the cal and save the calsets

For i = 1 To chans

    calset(i - 1) = scpi.Parse("SENS" & CStr(i+1) & ":CORR:COLL:GUID:SAVE ON")

Next

MsgBox ("SMC Cals Complete!")

```

Last Modified:

15-Mar-2011 Modified for FCA2
6-Feb-2008 MX New topic

Create New Cal Kit using SCPI

When creating new cal kits programmatically, the order in which cal kit commands are sent can be important.

For example to create a kit with opens, shorts, loads, and thrus. Be sure to use the following sequence for each newly defined standard.

1. Programmatically select the standard number
2. Programmatically select the standard type.
3. Program the cal standard's values.
4. Repeat steps 1, 2, 3 for additional new standards being defined.

```
10  !
20  !
30  ! This example program demonstrates how to create
40  ! new PNA calibration kits.
50  !
60  ! 1) Select a kit not previously defined
70  ! 2) Define open, short, load, and thru cal standards
80  ! Note: Each of the newly defined standards is assigned
90  ! a default connector name. These default connector names
100 ! will be replaced in subsequent steps.
110 ! 3) Use the delete connector command to remove default
120 ! connector names.
130 ! 4) Add connectors. Specify:
140 !     Start and Stop Freq
150 !     Z - Impedance
160 !     sex - MALE, FEMALE, NONE
170 !     media - COAX, WAVE
180 !     cutoff - Frequency for waveguide
190 ! 5) Assign the appropriate connector to each standard
200 ! 6) Modify the class assignments for the standards defined
210 ! 7) Verify the kit values
220 !
230 ! Additional Note: After setting each new cal kit value, it is
240 ! recommended that the program periodically perform queries to
250 ! verify the new values.
260 !
270 ! This will prevent program synchronization issues that can
affect
280 ! final values stored within new cal kits.
290 !
```

```

300  !-----
310  !
320  ! Set up I/O path
330  ASSIGN @Na TO 716
340  DIM Calkname$(80),Conn$(80)
350  INTEGER Calkitnum
360  !
370  CLEAR SCREEN
380  !
390  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
400  ! Designate the kit selection to be used for performing cal's
410  OUTPUT @Na;":sens:corr:ckit:count?"
420  ENTER @Na;Calkitnum
430  Calkitnum=Calkitnum+1
440  OUTPUT @Na;":sens:corr:coll:ckit "&VAL$(Calkitnum)
450  !
460  ! Name this kit with your own name
470  OUTPUT @Na;":sens:corr:coll:ckit:name ""Special 2.4 mm Model
85056""
480  !
490  !
500  DISP "Defining kit std 1..."
510  ! Now set up standard #1
520  OUTPUT @Na;":sens:corr:coll:ckit:stan 1"
530  OUTPUT @Na;":sens:corr:coll:ckit:stan:type SHORT"
540  Get_std
550  OUTPUT @Na;":sens:corr:coll:ckit:stan:char coax"
560  OUTPUT @Na;":sens:corr:coll:ckit:stan:label ""My Short""
570  Get_label
580  !
590  DISP "Defining kit std 2..."
600  ! Now set up standard #2
610  OUTPUT @Na;":sens:corr:coll:ckit:stan 2"
620  OUTPUT @Na;":sens:corr:coll:ckit:stan:type OPEN"
630  Get_std
640  OUTPUT @Na;":sens:corr:coll:ckit:stan:char coax"
650  OUTPUT @Na;":sens:corr:coll:ckit:stan:label ""My Open""
660  Get_label
670  !
680  DISP "Defining kit std 3..."
690  ! Now set up standard #3
700  OUTPUT @Na;":sens:corr:coll:ckit:stan 3"
710  OUTPUT @Na;":sens:corr:coll:ckit:stan:type LOAD"
720  Get_std

```

```
730 OUTPUT @Na;":sens:corr:coll:ckit:stan:char coax"
740 OUTPUT @Na;":sens:corr:coll:ckit:stan:label "My Fixed Load""
750 Get_label
760 !
770 DISP "Defining kit std 4..."
780 ! Now set up standard #4
790 OUTPUT @Na;":sens:corr:coll:ckit:stan 4"
800 OUTPUT @Na;":sens:corr:coll:ckit:stan:type THRU"
810 Get_std
820 OUTPUT @Na;":sens:corr:coll:ckit:stan:char coax"
830 OUTPUT @Na;":sens:corr:coll:ckit:stan:label "My Thru""
840 Get_label
850 !
860 DISP "Defining kit std 5..."
870 ! Now set up standard #5
880 OUTPUT @Na;":sens:corr:coll:ckit:stan 5"
890 OUTPUT @Na;":sens:corr:coll:ckit:stan:type SLOAD"
900 Get_std
910 OUTPUT @Na;":sens:corr:coll:ckit:stan:char coax"
920 OUTPUT @Na;":sens:corr:coll:ckit:stan:label "Sliding Load""
930 Get_label
940 !
950 DISP "Defining kit std 6..."
960 ! Now set up standard #6
970 !
980 OUTPUT @Na;":sens:corr:coll:ckit:stan 6"
990 OUTPUT @Na;":sens:corr:coll:ckit:stan:type SHORT"
1000 Get_std
1010 OUTPUT @Na;":sens:corr:coll:ckit:stan:char coax"
1020 OUTPUT @Na;":sens:corr:coll:ckit:stan:label "Short""
1030 Get_label
1040 !
1050 DISP "Defining kit std 7..."
1060 ! Now set up standard #7
1070 OUTPUT @Na;":sens:corr:coll:ckit:stan 7"
1080 OUTPUT @Na;":sens:corr:coll:ckit:stan:type SHORT"
1090 Get_std
1100 OUTPUT @Na;":sens:corr:coll:ckit:stan:char coax"
1110 OUTPUT @Na;":sens:corr:coll:ckit:stan:label "Short""
1120 Get_label
1130 !
1140 DISP "Defining kit std 8..."
1150 ! Now set up standard #8
1160 !
```

```
1170 OUTPUT @Na;":sens:corr:coll:ckit:stan 8"
1190 OUTPUT @Na;":sens:corr:coll:ckit:stan:type ARBI"
1200 Get_std
1210 OUTPUT @Na;":sens:corr:coll:ckit:stan:char coax"
1220 OUTPUT @Na;":sens:corr:coll:ckit:stan:TZR 15;"
1230 OUTPUT @Na;":sens:corr:coll:ckit:stan:TZI -9;"
1240 OUTPUT @Na;":sens:corr:coll:ckit:stan:label ""Z Load""
1250 Get_label
1260 !
1270 !
1280 !
1290 ! First remove any old connector names
1300 OUTPUT @Na;":sens:corr:coll:ckit:conn:del"
1310 ! Verify that no connectors are currently installed
1320 OUTPUT @Na;":sens:corr:coll:ckit:conn:cat?"
1330 ENTER @Na;Conn$
1340 PRINT "Verify empty list: ";Conn$
1350 !
1360 ! Define your new connectors
1370 OUTPUT @Na;":sens:corr:coll:ckit:conn:add ""PSC
2.4"" ,0HZ,999GHZ,50.0,MALE,COAX,0.0"
1380 OUTPUT @Na;":sens:corr:coll:ckit:conn:add ""PSC
2.4"" ,0HZ,999GHZ,50.0,FEMALE,COAX,0.0"
1390 !
1400 ! Verify that the new connectors are installed
1410 OUTPUT @Na;":sens:corr:coll:ckit:conn:cat?"
1420 ENTER @Na;Conn$
1430 PRINT "Verify new connectors: ";Conn$
1440 DISP ""
1450 !
1460 DISP "Defining conn std 1..."
1470 ! Now set up standard #1
1480 OUTPUT @Na;":sens:corr:coll:ckit:stan 1"
1490 Verify_std
1500 OUTPUT @Na;":sens:corr:coll:ckit:conn:snam ""PSC
2.4"" ,FEMALE,1"
1510 Print_connector
1520 !
1530 DISP "Defining conn std 2..."
1540 ! Now set up standard #2
1550 OUTPUT @Na;":sens:corr:coll:ckit:stan 2"
1560 Verify_std
1570 OUTPUT @Na;":sens:corr:coll:ckit:conn:snam ""PSC
2.4"" ,FEMALE,1"
```

```
1580 Print_connector
1590 !
1600 DISP "Defining conn std 3..."
1610 ! Now set up standard #3
1620 OUTPUT @Na;":sens:corr:coll:ckit:stan 3"
1630 Verify_std
1640 OUTPUT @Na;":sens:corr:coll:ckit:conn:snam ""PSC
2.4"" ,FEMALE,1"
1650 Print_connector
1660 !
1670 DISP "Defining conn std 4..."
1680 ! Now set up standard #4
1690 OUTPUT @Na;":sens:corr:coll:ckit:stan 4"
1700 Verify_std
1710 OUTPUT @Na;":sens:corr:coll:ckit:conn:snam ""PSC
2.4"" ,FEMALE,1"
1720 OUTPUT @Na;":sens:corr:coll:ckit:conn:snam ""PSC 2.4"" ,MALE,2"
1730 Print_connector
1740 !
1750 DISP "Defining conn std 5..."
1760 ! Now set up standard #5
1770 OUTPUT @Na;":sens:corr:coll:ckit:stan 5"
1780 OUTPUT @Na;":sens:corr:coll:ckit:stan:label ""Sliding Load""
1790 Verify_std
1800 OUTPUT @Na;":sens:corr:coll:ckit:conn:snam ""PSC 2.4"" ,MALE,1"
1810 Print_connector
1820 !
1830 DISP "Defining conn std 6..."
1840 ! Now set up standard #6
1850 !
1860 OUTPUT @Na;":sens:corr:coll:ckit:stan 6"
1870 Verify_std
1880 OUTPUT @Na;":sens:corr:coll:ckit:conn:snam ""PSC 2.4"" ,MALE,1"
1890 Print_connector
1900 !
1910 DISP "Defining conn std 7..."
1920 ! Now set up standard #7
1930 OUTPUT @Na;":sens:corr:coll:ckit:stan 7"
1940 Verify_std
1950 OUTPUT @Na;":sens:corr:coll:ckit:conn:snam ""PSC 2.4"" ,MALE,1"
1960 Print_connector
1970 !
1980 DISP "Defining conn std 8..."
1990 ! Now set up standard #8
```

```
2000 OUTPUT @Na;":sens:corr:coll:ckit:stan 8"
2010 Verify_std
2020 OUTPUT @Na;":sens:corr:coll:ckit:conn:snam ""PSC 2.4"",MALE,1"
2030 Print_connector
2040 !
2050 DISP "Class assignments..."
2060 !
2070 ! Designate the "order" associated with measuring the standards
2080 !
2090 ! Set Port 1, 1st standard measured to be standard #2
2100 OUTPUT @Na;":sens:corr:coll:ckit:order1 2"
2110 ! Set Port 1, 2nd standard measured to be standard #1
2120 OUTPUT @Na;":sens:corr:coll:ckit:order2 1,6,7"
2130 ! Set Port 1, 3rd standard measured to be standard #3 and #5
2140 OUTPUT @Na;":sens:corr:coll:ckit:order3 3,5"
2150 ! Set Port 1, 4th standard measured to be standard #4
2160 OUTPUT @Na;":sens:corr:coll:ckit:order4 4"
2170 !
2180 ! Set Port 2, 1st standard measured to be standard #2
2190 OUTPUT @Na;":sens:corr:coll:ckit:order5 2"
2200 ! Set Port 2, 2nd standard measured to be standard #1
2210 OUTPUT @Na;":sens:corr:coll:ckit:order6 1,6,7"
2220 ! Set Port 2, 3rd standard measured to be standard #3 and #6
2230 OUTPUT @Na;":sens:corr:coll:ckit:order7 3,5"
2240 ! Set Port 2, 4th standard measured to be standard #4
2250 OUTPUT @Na;":sens:corr:coll:ckit:order8 4"
2260 !
2270 ! Set Port 1, 1st standard
2280 OUTPUT @Na;":sens:corr:coll:ckit:olabel1 ""MyOpen1""
2290 ! Set Port 1, 2nd standard
2300 OUTPUT @Na;":sens:corr:coll:ckit:olabel2 ""MyShorts1""
2310 ! Set Port 1, 3rd standard
2320 OUTPUT @Na;":sens:corr:coll:ckit:olabel3 ""MyLoads1""
2330 ! Set Port 1, 4th standard measured to be standard #4
2340 OUTPUT @Na;":sens:corr:coll:ckit:olabel4 ""MyThru1""
2350 !
2360 ! Set Port 2, 1st standard
2370 OUTPUT @Na;":sens:corr:coll:ckit:olabel5 ""MyOpen2""
2380 ! Set Port 2, 2nd standard
2390 OUTPUT @Na;":sens:corr:coll:ckit:olabel6 ""MyShorts2""
2400 ! Set Port 2, 3rd standard
2410 OUTPUT @Na;":sens:corr:coll:ckit:olabel7 ""MyLoads2""
2420 ! Set Port 2, 4th standard
2430 OUTPUT @Na;":sens:corr:coll:ckit:olabel8 ""MyThrus2""
```

```
2440 !
2450 BEEP
2460 DISP "Done!"
2470 END
2480 SUB Get_label
2490     OUTPUT 716;":sens:corr:coll:ckit:stan:label?"
2500     ENTER 716;Label$
2510     PRINT Label$
2520 SUBEND
2530 !
2540 SUB Get_std
2550     OUTPUT 716;":sens:corr:coll:ckit:stan:type?"
2560     ENTER 716;Type$
2570     PRINT Type$
2580 SUBEND
2590 !
2600 SUB Print_connector
2610     DIM Nam$(40)
2620     OUTPUT 716;":sens:corr:coll:ckit:conn:sname?"
2630     ENTER 716;Nam$
2640     PRINT Nam$
2650 SUBEND
2660 !
2670 SUB Verify_std
2680     OUTPUT 716;":sens:corr:coll:ckit:stan:label?"
2690     ENTER 716;Label$
2700 SUBEND
2710 !
```

Create a Custom Power Meter Driver

This topic requires that you have a working knowledge of Visual Basic.

This topic will help you create your own power meter driver for use with Source Power Calibration on the PNA. If you are using an Agilent Power Meter to perform a Source Power Calibration, you do NOT need to create your own driver.

Your Power Meter driver will be created from a template written in Visual Basic using VISA over the GPIB bus.

Note: This procedure applies to Visual Basic 6.0. Applicability to Visual Basic .NET has not yet been investigated.

- [Prepare Template Files](#)
- [Modify Template Files](#)
- [Compile, Copy, and Register, Your New Driver](#)
- [Test Your new Driver](#)

Other SCPI Example Programs

Prepare Template Files

1. Copy all the files from the PNA hard drive C:/Program Files/Agilent/Network Analyzer/Automation/Power Meter Driver Template folder, to a folder on your development PC.
2. In Visual Basic click **File**, then **Open Project...**, find **MyPowerMeter.vbp** (a file you copied from the PNA). Click **Open**. This is a VB ActiveX EXE template, which you will fill in to become your driver.
3. Click **Project**, then **MyPowerMeter Properties**. Click the **General** tab.
4. Overwrite the Project Name with a name of your own choosing. This will be the name of your driver's type library (also the default name of your exe).

Note If the name of your exe does not match the VB Project Name with which it was compiled, registration of the exe on the PNA will not succeed.

5. Set the Project Description. After building your driver if you wish to test it using VB, this is the string that will show up in the VB References list of your test project, and also in the lower pane of the VB Object Browser.
6. Set the Thread Pool size to 1 thread.

7. Click **OK** to close the project properties dialog.
8. From the VB **Project** menu, click **References...** Ensure that **Agilent PNA Power Meter 1.0 Type Library** and **VISA Library** are checked. Click **OK**.

Note: Agilent's implementation of VISA is installed as part of the Agilent I/O Libraries on the PNA. For help on VISA, go to the Windows Start button on your PNA, select Programs, Agilent IO Libraries, VISA Help.

Modify Template Files

From Visual Basic **View** menu click **Project Explorer**. Expand the **Modules** and **Class Modules** folders. Ensure there is one module (WinAPI) and one class module (PowerMeter).

Let's look at the WinAPI module first.

1. In the **Project Explorer** window, click **WinAPI**.
2. From the **View** menu click **Code**.

There is only one line of code you should need to modify in this module: the value of the string constant named `SIDSEARCH`. The comments preceding the declaration of that string describe how to change it. The rest of this module contains functions which will use the Microsoft Windows API to insure proper registration of your driver on the PNA. If you know of other Windows API functions you feel might be helpful to call from within your PowerMeter class module (to help in formatting data, for example), this module would be the place to declare them.

Now let's look at the class module.

1. In the Project Explorer window, click **PowerMeter**.
2. From the **View** menu click **Properties Window**. The **Instancing** property must be set to MultiUse. This allows other applications to create objects from this class, such that one instance of your driver EXE can supply more than one such object at a time.
3. From the **View** menu click **Code**.

Do NOT modify the Interfaces to IPowerMeter subroutines and functions. PNA source power cal expects to find these interfaces as they are currently defined.

The only members that you need to supply code to are those containing "**Your code here**" comments.

In addition, comments have been provided at the beginning of each member to describe the information that member needs to be read from or written to the power meter.

To get an idea of how communicate with the power meter using the VISA functions **viWrite** and

viRead, examine the code which has been implemented for you in `IPowerMeter_Connect`, `IPowerMeter_QueryMeter`, and `IPowerMeter_WriteMeter`.

Compile, Copy, and Register Your New Driver

When your driver is ready to run, you will first need to compile it into an EXE.

From the File menu select **Make exe**.

After compiling, the following will instruct VB to use the same ID (GUID) every time you re-compile your project.

1. From the **Project** menu, click **PowerMeter Properties**.
2. On the **Component** tab, select **Binary Compatibility** and click ...
3. Browse to and select your project EXE. Click **Open**.
4. Click **OK** to close **Project Properties**.
5. Save your project.
6. Copy your driver EXE file to a folder on your PNA (do NOT use `C:/Program Files/Agilent/Network Analyzer/Automation/Power Meter Driver Template` folder).
7. Run the EXE file. A message box will pop up reporting whether or not registration was successful. If not successful, it will make a suggestion on what to fix.

When your driver is properly registered, PNA Source Power Cal should be able to associate it with the ID string of your power meter.

Test Your Power Meter Driver

We have also provided a Visual Basic project to test your new Power Meter driver. This project individually calls every `IPowerMeter` method and property in your driver to verify that it performs correctly. Before running the test your PC and PNA must be configured to communicate using DCOM.

1. Connect your PC and the PNA to LAN.
2. Add your PC logon to the PNA. Both logons and password must match to communicate using DCOM. See [Additional PNA users](#).
3. Configure your driver using DCOM Config on the PNA. This will give you permission to launch and access the driver. See [Configure for COM-DCOM Programming](#).

Modify the Test Project

1. In Visual Basic click **File**, then **Open Project...**, find **MyPowerMeterTest.vbp** (a file you copied from the PNA). Click **Open**.
2. From the **Project** menu, click **References...** From the list, find and check your new Power Meter Driver. (It should have been registered on your PC when you successfully made your driver EXE.) Click **OK**.
3. From the **View** menu click **Code**.
4. Modify the **CreateObject** line as follows:
Replace **MyPowerMeter** with the Project Name that you chose for your driver
Replace **MyPNA** with the Computer Name of your PNA.
For example:

```
Set PowerMeterObj = CreateObject("AcmeBrand.PowerMeter", "AGILENT-PNA123")
```

(This assumes that you kept **PowerMeter** as class module name in your driver.)

Run the Test Project

Ensure your power meter is connected to the PNA with a GPIB cable.

Put the PNA in system controller mode:

1. From the PNA **System** menu point to **Configure** then click **SICL/GPIB**.
2. In the GPIB box click **System Controller**.

Run the test project. If there are no errors, the driver is created successfully. If there are errors, try to figure out what went wrong and fix it. Then re-compile, re-copy the .exe to the PNA, and re-run the test. You should not need to re-register the driver or re-modify the test program.

ECALConfidence Check using SCPI

This Visual Basic program performs a complete ECAL confidence check.

To run this program, you need:

- An established GPIB interface connection
- Agilent's VISA or National Instrument's VISA installed on your PC
- The module visa32.bas added to your VB project.
- A form with two buttons: cmdRun and cmdQuit
- A calibrated S11 1-port or N-port measurement active on Channel 1
- Window 1 is visible

Note: A confidence check can NOT be performed remotely from User Characterizations that are stored on the PNA disk.

[See Other SCPI Example Programs](#)

```
'Session to VISA Default Resource Manager
Private defRM As Long
'Session to PNA
Private viPNA As Long
'VISA function status return code
Private status As Long

Private Sub Form_Load()
    defRM = 0
End Sub

Private Sub cmdRun_Click()
'String to receive data from the PNA
Dim strReply As String * 200

' Open the VISA default resource manager
status = viOpenDefaultRM(defRM)
If (status < VI_SUCCESS) Then HandleVISAError

' Open a VISA session (viPNA) to the PNA at GPIB address 16.
status = viOpen(defRM, "GPIB0::16::INSTR", 0, 0, viPNA)
If (status < VI_SUCCESS) Then HandleVISAError

' Need to set the VISA timeout value to give all our GPIB Reads
' sufficient time to complete before a timeout error occurs.
```

```

' For this example, let's try setting the limit to
' 10000 milliseconds (10 seconds).
status = viSetAttribute(vipNA, VI_ATTR_TMO_VALUE, 10000)
If (status < VI_SUCCESS) Then HandleVISAError

' Get the catalog of all the measurements currently on Channel 1.
status = myGPIBWrite(vipNA, "CALC1:PAR:CAT?")
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBRead(vipNA, strReply)
If (status < VI_SUCCESS) Then HandleVISAError

' If an S11 measurement named "MY_S11" doesn't already exist,
' then create it.
If InStr(strReply, "MY_S11") = 0 Then
    status = myGPIBWrite(vipNA, "CALC1:PAR:DEF:EXT MY_S11,S11")
    If (status < VI_SUCCESS) Then HandleVISAError
End If
strReply = ""

' Get the catalog of all the trace numbers currently active
' in Window 1.
status = myGPIBWrite(vipNA, "DISP:WIND1:CAT?")
If (status < VI_SUCCESS) Then HandleVISAError

status = myGPIBRead(vipNA, strReply)
If (status < VI_SUCCESS) Then HandleVISAError

' If a trace number 4 already exists in Window 1, then this
' will remove it.
If InStr(strReply, "4") > 0 Then
    status = myGPIBWrite(vipNA, "DISP:WIND1:TRAC4:DEL")
    If (status < VI_SUCCESS) Then HandleVISAError
End If

' Set trace number 4 to MY_S11.
status = myGPIBWrite(vipNA, "DISP:WIND1:TRAC4:FEED MY_S11")
If (status < VI_SUCCESS) Then HandleVISAError

' Set up trace view so we are viewing only the data trace.
status = myGPIBWrite(vipNA, "DISP:WIND1:TRAC4 ON")
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBWrite(vipNA, "DISP:WIND1:TRAC4:MEM OFF")
If (status < VI_SUCCESS) Then HandleVISAError

' Select MY_S11 as the measurement to be used for the
' Confidence Check.
status = myGPIBWrite(vipNA, "SENS1:CORR:CCH:PAR MY_S11")
If (status < VI_SUCCESS) Then HandleVISAError

' Acquire the S11 confidence check data from ECal Module A
' into the memory buffer (asking for an OPC reply when it's done).
status = myGPIBWrite(vipNA, "SENS1:CORR:CCH:ACQ ECAL1;*OPC?")

```

```

If (status < VI_SUCCESS) Then HandleVISAError

' The PNA sends an OPC reply ("+1") when the confidence data
' acquisition into memory is complete, so this Read is waiting on
' the reply until it is received.
status = myGPIBRead(viPNA, strReply)
If (status < VI_SUCCESS) Then HandleVISAError

' Turn on trace math so the trace shows data divided by memory.
' You can be confident the S11 calibration is reasonably good if
' the displayed trace varies no more than a few tenths of a dB
' from 0 dB across the entire span.
status = myGPIBWrite(viPNA, "CALC1:PAR:SEL MY_S11")
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBWrite(viPNA, "CALC1:MATH:FUNC DIV")
If (status < VI_SUCCESS) Then HandleVISAError
End Sub

Private Sub cmdQuit_Click()
' Turn off trace math
status = myGPIBWrite(viPNA, "CALC1:MATH:FUNC NORM")
If (status < VI_SUCCESS) Then HandleVISAError

' Conclude the confidence check to set the ECal module
' back to it's idle state.
status = myGPIBWrite(viPNA, "SENS1:CORR:CCH:DONE")
If (status < VI_SUCCESS) Then HandleVISAError

' Close the resource manager session (which also closes
' the session to the PNA).
If defRM <> 0 Then Call viClose(defRM)

' End the program
End
End Sub

Private Function myGPIBWrite(ByVal viHandle As Long, ByVal strOut As String) As Long
' The "+ Chr$(10)" appends an ASCII linefeed character to the output, for
' terminating the write transaction.
myGPIBWrite = viVPrintf(viHandle, strOut + Chr$(10), 0)
End Function

Private Function myGPIBRead(ByVal viHandle As Long, strIn As String) As Long
myGPIBRead = viVScanf(viHandle, "%t", strIn)
End Function

Sub HandleVISAError()
Dim strVisaErr As String * 200
Call viStatusDesc(defRM, status, strVisaErr)
MsgBox "**** Error : " + strVisaErr, vbExclamation
End
End Sub

```

Establish a VISA Session

This Visual Basic program demonstrates how to send a SCPI command using VISA and the Agilent IO libraries. To run this program, you need:

- Your PC and PNA both connected to a LAN (for communicating with each other).
- The SICL and VISA components of Agilent's I/O Libraries software installed on your PC. Both are included when you install the software, unless you already have another vendor's VISA installed. Then specify Full SICL and VISA installation to overwrite the other vendor's VISA.
- The module visa32.bas added to your VB project. After you install VISA, the module will be located at C:\XIPNP\WINNT (or equivalent)\INCLUDE\Visa32.bas
- A form with two buttons: cmdRun and cmdQuit.
- Your PC configured to be a VISA LAN Client, and the SICL Server capability enabled on the PNA. See [Configure for VISA and SICL](#)

[See Other SCPI Example Programs](#)

Note: This example is a piece of a larger VISA program that [performs a source power calibration](#).

```
'Session to VISA Default Resource Manager
Private defRM As Long
'Session to PNA
Private viPNA As Long
'VISA function status return code
Private status As Long

Private Sub Form_Load()
defRM = 0
End Sub

Private Sub cmdRun_Click()
' String to receive data from the PNA.
' Dimensioned large enough to receive scalar comma-delimited values
' for 21 frequency points (20 ASCII characters per point)
Dim strReply As String * 420

' Open the VISA default resource manager
status = viOpenDefaultRM(defRM)
If (status < VI_SUCCESS) Then HandleVISAError

' Open a VISA session (viPNA) to the SICL LAN server
' at "address 16" on the PNA pointed to by the "GPIB0"
' VISA LAN Client on this PC.
' CHANGE GPIB0 TO WHATEVER YOU PNA IS SET TO
status = viOpen(defRM, "GPIB0::16::INSTR", 0, 0, viPNA)
If (status < VI_SUCCESS) Then HandleVISAError
```

```

' Need to set the VISA timeout value to give all our calls to
' myGPIBRead sufficient time to complete before a timeout
' error occurs.
' For this example, let's try setting the limit to
' 30000 milliseconds (30 seconds).
status = viSetAttribute(viPNA, VI_ATTR_TMO_VALUE, 30000)
If (status < VI_SUCCESS) Then HandleVISAError

' Preset the PNA
status = myGPIBWrite(viPNA, "SYST:PRES")
If (status < VI_SUCCESS) Then HandleVISAError

' Print the data using a message box
MsgBox strReply
End Sub

Private Sub cmdQuit_Click()
' Close the resource manager session (which also closes
' the session to the PNA).
If defRM <> 0 Then Call viClose(defRM)

' End the program
End
End Sub

Private Function myGPIBWrite(ByVal viHandle As Long, ByVal strOut As String) As Long
' The "+ Chr$(10)" appends an ASCII linefeed character to the
' output, for terminating the write transaction.
myGPIBWrite = viPrintf(viHandle, strOut + Chr$(10), 0)
End Function

Private Function myGPIBRead(ByVal viHandle As Long, strIn As String) As Long
myGPIBRead = viVscanf(viHandle, "%t", strIn)
End Function

Sub HandleVISAError()
Dim strVisaErr As String * 200
Call viStatusDesc(defRM, status, strVisaErr)
MsgBox "*** Error : " + strVisaErr, vbExclamation
End
End Sub

```

External Test Set Control using SCPI

This program demonstrates the use of several External Test Set Control commands.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as ExtTS.vbs. [Learn how to setup and run the macro.](#)

```
' Demonstrate some SCPI commands for external testsets.

Dim pna

Set pna = CreateObject("AgilentPNA835x.Application")

Set scpi = pna.ScpiStringParser

' The K64 testset is only usable on a 4-port PNA

If (pna.NumberOfPorts <> 4) Then

MsgBox("This program only runs on 4-port analyzers.")

Else

'If Help is active, show the measurement window and help

scpi.Execute("DISP:ARR TILE")

'Return the list of supported test sets

list=scpi.Execute("SENS:MULT:CATalog?")

MsgBox(list)

'***** K64 *****

'The K64 is connected using the Testset I/O

'connector. There is no handshake information.

'Therefore, a testset need not be connected.

' Load a configuration file.

scpi.Execute("SENS:MULT1:TYPE 'Z5623AK64'")

scpi.Execute("SENS:MULT1:ADDR 0")

'return stuff about the test set
```

```
' Returns number of input ports
Inports=scpi.Execute("SENS:MULT1:INCount?")
MsgBox("Input Ports: " & CStr(Inports))

' Returns number of output ports
ports=scpi.Execute("SENS:MULT1:COUNT?")
MsgBox("Output Ports: " & CStr(ports))

' Returns valid output ports for each input port
For portNum = 1 To Inports
ports=scpi.Execute("SENS:MULT1:PORT" & CStr(portNum) & ":CAT?")
MsgBox("Port " & CStr(portNum) & " catalog: " & (ports))
Next

'Set different port mapping
scpi.Execute("SENS:MULT1:ALLPorts '1 ext R,2 ext R,3 ext R,4 ext R'")

'Return port mapping
portMap=scpi.Execute("SENS:MULT1:ALLPorts?")
MsgBox("Ports will be mapped to " & CStr(portMap))

' Enable external testset control and execute port mapping. This automatically
enables status bar display as well.
scpi.Execute("SENS:MULT1:STATE 1")
MsgBox("Z5623A K64 Enabled")
End If
```

Getting and Putting Data using SCPI

This Visual Basic Program does the following:

- Reads data from the analyzer
- Puts the data back into memory
- To see the data on the analyzer after running the program, from the front panel click:
Trace - Math/Memory - Memory Trace

To run this program, you need:

- An established [GPIB interface connection](#)

[See Other SCPI Example Programs](#)

Note: To change the read and write location of data, removing the comment from the beginning of ONE of the lines, and replace the comment in the beginning of the SDATA and SMEM lines.

```
Private Sub ReadWrite_Click()
Dim i As Integer
Dim t As Integer
Dim q As Integer
Dim dat As String
Dim cmd As String
Dim datum() As Double

GPIB.Configure
GPIB.Write "SYSTem:PRESet;*wai"

'Select the measurement
GPIB.Write "CALCulate:PARAMeter:SElect 'CH1_S11_1'"

'Read the number of data points
GPIB.Write "SENSe1:SWEep:POIN?"
numpts = GPIB.Read

'Turn continuous sweep off
GPIB.Write "INITiate:CONTinuous OFF"

'Take a sweep
GPIB.Write "INITiate:IMMEDIATE;*wai"

'Ask for the Data

'PICK ONE OF THESE LOCATIONS TO READ
'GPIB.Write "CALCulate:DATA? FDATA" 'Formatted Meas
'GPIB.Write "CALCulate:DATA? FMEM" 'Formatted Memory
GPIB.Write "CALCulate:DATA? SDATA" 'Corrected, Complex Meas
'GPIB.Write "CALCulate:DATA? SMEM" 'Corrected, Complex Memory
'GPIB.Write "CALCulate:DATA? SCORR1" 'Error-Term Directivity
```

```
'Number of values returned per data point
'q = 1 ' Pick this if reading FDATA or FMEM
q = 2 ' Otherwise pick this

'Parse the data
ReDim datum(q, numpts)
For i = 0 To numpts - 1
  For t = 0 To q - 1
    'Read the Data
    dat = GPIB.Read(20)
    'Parse it into an array
    datum(t, i) = Val(dat)
  Next t
Next i

'PUT THE DATA BACK IN
GPIB.Write "format ascii"

'PICK ONE OF THESE LOCATIONS TO PUT THE DATA
'cmd = "CALCulate:DATA FDATA," 'Formatted Meas
'cmd = "CALCulate:DATA FMEM," 'Formatted Memory
'cmd = "CALCulate:DATA SDATA," 'Corrected, Complex Meas
cmd = "CALCulate:DATA SMEM," 'Corrected, Complex Memory
'cmd = "CALCulate:DATA SCORR1," 'Error-Term Directivity

For i = 0 To numpts - 1
  For t = 0 To q - 1
    If i = numpts - 1 And t = q - 1 Then
      cmd = cmd & Format(datum(t, i))
    Else
      cmd = cmd & Format(datum(t, i)) & ","
    End If
  Next t
Next i

GPIB.Write cmd
End Sub
```

GPIB Pass-Through Example

The SCPI [SYSTEM](#) commands used in this example allow you to send GPIB commands to another GPIB device through the PNA. The other device would typically be connected to the PNA through the System Controller GPIB port on the PNA rear-panel or alternatively be connected using a [USB/GPIB interface](#). Uncomment the line in [Blue text](#) in the example to open a session for a USB/GPIB interface.

This VB Script example uses the COM [SCPIStringParser](#) object. However, this is not critical to the use of these commands; they can be sent using the normal syntax of your programming environment. Using the SCPIStringParser over LAN allows you to communicate with GPIB devices without requiring your remote PC to have a GPIB interface card installed.

Although this method of pass-through works for most applications, there are a couple of limitations:

- All data is transferred using ASCII format. Therefore, transferring large blocks of data is very slow.
- Only read and write functions are possible. Service Interrupts are not supported.

[See Other SCPI Example Programs](#)

```
option explicit
dim app
set app = CreateObject("AgilentPNA835x.Application")

dim p
set p = app.ScpiStringParser

' Open a new GPIB session on Bus:0 Device:14 Timeout: 100ms
p.Parse "SYST:COMM:GPIB:RDEV:OPEN 0,14,100"
' The following commented-out line shows opening the same session but
' for a USB/GPIB interface with VISA interface number GPIB4
'p.Parse "SYST:COMM:GPIB:RDEV:OPEN 4,14,100"
dim handleAsStr

' Retrieve the handle (ID number)
handleAsStr = p.Parse ("SYST:COMM:GPIB:RDEV:OPEN?")

' Convert the handle to an integer
dim handleAsInt
handleAsInt = CInt(handleAsStr)

' Send the "*IDN?" query
p.Parse "SYST:COMM:GPIB:RDEV:WRITE " & handleAsInt & ", '*IDN?'"

' Read its results
dim idn
idn = p.Parse("SYST:COMM:GPIB:RDEV:READ? " & handleAsInt)
msgbox idn
```

```
' Close the GPIB session
```

```
p.Parse "SYST:COMM:GPIB:RDEV:CLOSE " & handleAsInt
```

Last modified:

30-May-2012 Added uncomment line and other edit (BH)

GPIB using Visual C++

[See Other SCPI Example Programs](#)

```
/*
 * This example assumes the user's PC has a National Instruments GPIB board. The
example is comprised of three basic parts:
 *
 * 1. Initialization
 * 2. Main Body
 * 3. Cleanup
 *
 * The Initialization portion consists of getting a handle to the PNA and then doing
a GPIB clear of the PNA.
 *
 * The Main Body consists of the PNA SCPI example.
 *
 * The last step, Cleanup, releases the PNA for front panel control.
 */

#include <stdio.h>
#include <stdlib.h>

/*
 * Include the WINDOWS.H and DECL-32.H files. The standard Windows
 * header file, WINDOWS.H, contains definitions used by DECL-32.H and
 * DECL-32.H contains prototypes for the NI GPIB routines and constants.
 */
#include <windows.h>
#include "decl-32.h"

#define ERRMSGSIZE 1024 // Maximum size of SCPI command string
#define ARRAYSIZE 1024 // Size of read buffer

#define BDINDEX 0 // Board Index of GPIB board
#define PRIMARY_ADDR_OF_PNA 16 // GPIB address of PNA
#define NO_SECONDARY_ADDR 0 // PNA has no Secondary address
#define TIMEOUT T10s // Timeout value = 10 seconds
#define EOTMODE 1 // Enable the END message
#define EOSMODE 0 // Disable the EOS mode

int pna;
char ValueStr[ARRAYSIZE + 1];
char ErrorMnemonic[21][5] = {"EDVR", "ECIC", "ENOL", "EADR", "EARG",
    "ESAC", "EABO", "ENEB", "EDMA", "",
    "EOIP", "ECAP", "EFSO", "", "EBUS",
    "ESTB", "ESRQ", "", "", "", "ETAB"};

void GPIBWrite(char* SCPIcmd);
char *GPIBRead(void);
```

```

void GPIBCleanup(int Dev, char* ErrorMessage);

int main()
{
char *opc;
char *result;
char *value;

/*
 * =====
 * INITIALIZATION SECTION
 * =====
 */

/*
 * The application brings the PNA online using ibdev. A device handle,pna, is
returned and is used in all subsequent calls to the PNA.
 */
pna = ibdev(BDINDEX, PRIMARY_ADDR_OF_PNA, NO_SECONDARY_ADDR,
TIMEOUT, EOTMODE, EOSMODE);
if (ibsta & ERR)
{
printf("Unable to open handle to PNA/nibsta = 0x%x iberr = %d/n",
ibsta, iberr);
return 1;
}

/*
 * Do a GPIB Clear of the PNA. If the error bit ERR is set in ibsta, call
GPIBCleanup with an error message.
 */
ibclr (pna);
if (ibsta & ERR)
{
GPIBCleanup(pna, "Unable to perform GPIB clear of the PNA");
return 1;
}

/*
 * =====
 * MAIN BODY SECTION
 * =====
 */

// Reset the analyzer to instrument preset
GPIBWrite("SYSTEM:FPRESET");

// Create S11 measurement
GPIBWrite("CALCulatel:PARAMeter:DEFine:EXT 'My_S11',S11");

```

```

// Turn on Window #1
GPIBWrite("DISPlay:WINDow1:STATE ON");

// Put a trace (Trace #1) into Window #1 and 'feed' it from the measurement
GPIBWrite("DISPlay:WINDow1:TRACe1:FEED 'My_S11'");

// Setup the channel for single sweep trigger
GPIBWrite("INITiate1:CONTinuous OFF;*OPC?");
opc = GPIBRead();
GPIBWrite("SENSE1:SWEep:TRIGger:POINT OFF");

// Set channel parameters
GPIBWrite("SENSE1:SWEep:POINTs 11");
GPIBWrite("SENSE1:FREQuency:START 1000000000");
GPIBWrite("SENSE1:FREQuency:STOP 2000000000");

// Send a trigger to initiate a single sweep
GPIBWrite("INITiate1;*OPC?");
opc = GPIBRead();

// Must select the measurement before we can read the data
GPIBWrite("CALCulate1:PARAMeter:SElect 'My_S11'");

// Read the measurement data into the "result" string variable
GPIBWrite("FORMat ASCII");
GPIBWrite("CALCulate1:DATA? FDATA");
result = GPIBRead();

// Print the data to the display console window
printf("S11(dB) - Visual C++ SCPI Example for PNA/n/n");
value = strtok(result, ",");
while (value != NULL)
{
    printf("%s/n", value);
    value = strtok(NULL, ",");
}

/*
* =====
* CLEANUP SECTION
* =====
*/

/* The PNA is returned to front panel control. */
ibonl(pna, 0);

return 0;
}

/*
* Write to the PNA
*/

```

```

void GPIBWrite(char* SCPIcmd)
{
int length;
char ErrorMessage[ERRMSGSIZE + 1];
length = strlen(SCPIcmd) ;

    ibwrt (pna, SCPIcmd, length);
    if (ibsta & ERR)
    {
strcpy(ErrorMessage, "Unable to write this command to PNA:/n");
strcat(ErrorMessage, SCPIcmd);

GPIBCleanup(pna, ErrorMessage);
exit(1);
    }
}

/*
 * Read from the PNA
 */
char* GPIBRead(void)
{
    ibrd (pna, ValueStr, ARRAYSIZE);
    if (ibsta & ERR)
    {
GPIBCleanup(pna, "Unable to read from the PNA");
exit(1);
    }
else
    return ValueStr;
}

/*
 * After each GPIB call, the application checks whether the call succeeded. If an
NI-488.2 call fails, the GPIB driver sets the corresponding bit in the global status
variable. If the call failed, this procedure prints an error message, takes the PNA
offline and exits.
 */
void GPIBCleanup(int Dev, char* ErrorMessage)
{
    printf("Error : %s/nibsta = 0x%x iberr = %d (%s)/n",
ErrorMessage, ibsta, iberr, ErrorMnemonic[iberr]);
    if (Dev != -1)
    {
printf("Cleanup: Returning PNA to front panel control/n");
ibonl (Dev, 0);
    }
}

```

Load Error Terms during a Cal Sequence

This example requires that you already have a Cal Set named "foo" that contains a 1-port cal on port 1 and a 1-port cal on port 2.

This example starts a Guided Calibration specifying an Unknown Thru. It loads the 1-port Cals from the existing "foo" Cal Set, then recalculates the number of steps required to complete the cal. After loading the 1-port cals, only the Unknown Thru standard is left to acquire.

```
SENS:CORR:COLL:GUID:CONN:PORT1 "APC 3.5 female"
SENS:CORR:COLL:GUID:CONN:PORT2 "APC 3.5 female"
SENS:CORR:COLL:GUID:CKIT:PORT1 "85033D/E"
SENS:CORR:COLL:GUID:CKIT:PORT2 "85033D/E"
SENS:CORR:COLL:GUID:METH UNKN
' auto-create user calsets for SCPI
SENS:CORR:PREF:CSET:SAVU 1
SENS:CORR:COLL:GUID:INIT
' should return the number 7
SENS:CORR:COLL:GUID:STEPS?
' to port 1, from port 1 in calset
SENS:CORR:COLL:GUID:ETER:LOAD "foo",1,1
' to port 2, from port 2 in calset
SENS:CORR:COLL:GUID:ETER:LOAD "foo",2,2
' should now return the number 1
SENS:CORR:COLL:GUID:STEPS?
' measure the unknown thru
SENS:CORR:COLL:GUID:ACQ STAN1
' save the cal to new user calset
SENS:CORR:COLL:GUID:SAVE
```

Modify a Calibration Kit using SCPI

This Visual Basic program:

- Modifies Calibration kit number 3
- Completely defines standard #4 (thru)

To run this program, you need:

- An established [GPIB interface connection](#)

[See Other SCPI Example Programs](#)

```
'Modifying cal kit number 3
Calkitnum = 3

'Designate the kit selection to be used for performing cal's
GPIB.Write "SENSe:CORRection:COLLect:CKIT:SElect " & Val(Calkitnum)

'Reset to factory default values.
GPIB.Write "SENSe:CORRection:COLLect:CKIT:RESet " & Val(Calkitnum)

'Name this kit with your own name
GPIB.Write "SENSe:CORRection:COLLect:CKIT:NAME 'My Cal Kit'"

'Assign standard numbers to calibration classes
'Set Port 1, class 1 (S11A) to be standard #8
GPIB.Write "SENSe:CORRection:COLLect:CKIT:ORDeR1 8"
'Set Port 1, class 2 (S11B) to be standard #7
GPIB.Write "SENSe:CORRection:COLLect:CKIT:ORDeR2 7"
'Set Port 1, class 3 (S11C) to be standard #3
GPIB.Write "SENSe:CORRection:COLLect:CKIT:ORDeR3 3"
'Set Port 1, class 4 (S21T) to be standard #4
GPIB.Write "SENSe:CORRection:COLLect:CKIT:ORDeR4 4"
'Set Port 2, class 1 (S22A) to be standard #8
GPIB.Write "SENSe:CORRection:COLLect:CKIT:ORDeR5 8"
'Set Port 2, class 2 (S22B) to be standard #7
GPIB.Write "SENSe:CORRection:COLLect:CKIT:ORDeR6 7"
'Set Port 2, class 3 (S22C) to be standard #3
GPIB.Write "SENSe:CORRection:COLLect:CKIT:ORDeR7 3"
'Set Port 2, class 4 (S12T) to be standard #4
GPIB.Write "SENSe:CORRection:COLLect:CKIT:ORDeR8 4"

'Set up Standard #4 completely
'Select Standard #4; the rest of the commands act on it
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard 4"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:FMIN 300KHz"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:FMAX 9GHz"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:IMPedance 50"
```

```
GPIB.Write "SENSE:CORREction:COLLect:CKIT:STANdard:DELay 1.234 ns"  
GPIB.Write "SENSE:CORREction:COLLect:CKIT:STANdard:LOSS 23e6"  
GPIB.Write "SENSE:CORREction:COLLect:CKIT:STANdard:C0 0"  
GPIB.Write "SENSE:CORREction:COLLect:CKIT:STANdard:C1 1"  
GPIB.Write "SENSE:CORREction:COLLect:CKIT:STANdard:C2 2"  
GPIB.Write "SENSE:CORREction:COLLect:CKIT:STANdard:C3 3"  
GPIB.Write "SENSE:CORREction:COLLect:CKIT:STANdard:L0 10"  
GPIB.Write "SENSE:CORREction:COLLect:CKIT:STANdard:L1 11"  
GPIB.Write "SENSE:CORREction:COLLect:CKIT:STANdard:L2 12"  
GPIB.Write "SENSE:CORREction:COLLect:CKIT:STANdard:L3 13"  
GPIB.Write "SENSE:CORREction:COLLect:CKIT:STANdard:LAbel 'My Special Thru'"  
GPIB.Write "SENSE:CORREction:COLLect:CKIT:STANdard:TYPE THRU"  
GPIB.Write "SENSE:CORREction:COLLect:CKIT:STANdard:CHARacteristic Coax"
```

Perform a Guided 2-Port or 4-Port Cal using SCPI

This example performs a Guided 2-Port or 4-port Calibration using ONE set of calibration standards or an ECAL module.

A measurement must first be set up with desired frequency range, power, and so forth, ready to be calibrated.

The SCPI commands in this example are sent over a COM interface using the [SCPIStringParser](#) object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the PNA hard drive as *.vbs.

[Learn how to setup and run the macro.](#)

[See Guided Cal SCPI commands](#)

See Other SCPI Example Programs

```
Dim app
Dim scpi
' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

' To perform 2-port cal, Uncomment TwoPortGuidedCal()
' Then comment FourPortGuidedCal()

'Do 2-port Cal
'TwoPortGuidedCal()

'Do 4-port Cal
FourPortGuidedCal

Sub TwoPortGuidedCal()
' Select the connectors
scpi.Execute("sens:corr:coll:guid:conn:port1 ""APC 3.5 female"" ")
scpi.Execute("sens:corr:coll:guid:conn:port2 ""APC 3.5 male"" ")
scpi.Execute("sens:corr:coll:guid:conn:port3 ""Not used"" ")
scpi.Execute("sens:corr:coll:guid:conn:port4 ""Not used"" ")
MsgBox("Connectors defined for Ports 1 and 2")

' Select the Cal Kit for each port being calibrated.

scpi.Execute("sens:corr:coll:guid:ckit:port1 ""85052D"" ")
```

```

scpi.Execute("sens:corr:coll:guid:ckit:port2 ""85052D"" ")

' To use an ECal module instead, comment out the above two lines
' and uncomment the appropriate lines below:
' Your ECal module must already be connected
' via USB to the PNA.

'scpi.Parse "sens:corr:coll:guid:ckit:port1 'N4691-60004 ECal'"
'scpi.Parse "sens:corr:coll:guid:ckit:port2 'N4691-60004 ECal'"

' Non-factory characterizations are specified as follows:
'scpi.Parse "sens:corr:coll:guid:ckit:port2 'N4691-60004 User 1 ECal'"

' When two or more ECal modules with the same model number are connected
' also specify the serial number as follows:
'scpi.Parse "sens:corr:coll:guid:ckit:port2 'N4691-60004 ECal 01234'"

' When Disk Memory ECal user characterizations are used,
' specify both the User char and the serial number as follows:
'scpi.Parse "sens:corr:coll:guid:ckit:port2 'N4691-60004 MyDskChar ECal 01234'"

'

MsgBox("Cal kits defined for Ports 1 and 2")

' Initiate the calibration and query the number of steps
numSteps = GenerateSteps()
' Measure the standards, compute and apply the cal
MeasureAndComplete(numSteps)
End Sub

Sub FourPortGuidedCal()
' Select the connectors
scpi.Execute("sens:corr:coll:guid:conn:port1 ""APC 3.5 female"" ")
scpi.Execute("sens:corr:coll:guid:conn:port2 ""APC 3.5 female"" ")
scpi.Execute("sens:corr:coll:guid:conn:port3 ""APC 3.5 female"" ")
scpi.Execute("sens:corr:coll:guid:conn:port4 ""APC 3.5 female"" ")
MsgBox("Connectors defined for Ports 1 to 4")
' Select the Cal Kit for each port being calibrated.
scpi.Execute("sens:corr:coll:guid:ckit:port1 ""85052D"" ")
scpi.Execute("sens:corr:coll:guid:ckit:port2 ""85052D"" ")
scpi.Execute("sens:corr:coll:guid:ckit:port3 ""85052D"" ")
scpi.Execute("sens:corr:coll:guid:ckit:port4 ""85052D"" ")
' To use an ECal module instead, comment out the above four lines
' and uncomment these four lines and use the part number printed

```

```

' on your module (which in our case was N4431-60003), followed
' by the word 'ECal'. Your ECal module must already be connected
' via USB to the PNA.

' see above for ECal options

'scpi.Execute("sens:corr:coll:guid:ckit:port1 ""N4431-60003 ECal"" ")
'scpi.Execute("sens:corr:coll:guid:ckit:port2 ""N4431-60003 ECal"" ")
'scpi.Execute("sens:corr:coll:guid:ckit:port3 ""N4431-60003 ECal"" ")
'scpi.Execute("sens:corr:coll:guid:ckit:port4 ""N4431-60003 ECal"" ")
MsgBox("Cal kits defined for Ports 1 to 4")

' Initiate the calibration and query the number of steps
numSteps = GenerateSteps()
' If your selected cal kit is not a 4-port ECal module which can
' mate to all 4 ports at once, then you may want to choose which
' thru connections to measure for the cal. You must measure at
' least 3 different thru paths for a 4-port cal (for greatest
' accuracy you can choose to measure a thru connection for all 6
' pairings of the 4 ports). If you omit this command, the default
' is to measure from port 1 to port 2, port 1 to port 3, and
' port 1 to port 4. For this example we select to measure
' from port 1 to port 2, port 2 to port 3, and port 2 to port 4.
scpi.Execute("sens:corr:coll:guid:thru:ports 1,2,2,3,2,4")
' Re-generate the connection steps to account for the thru changes
numSteps = GenerateSteps()
' Measure the standards, compute and apply the cal
MeasureAndComplete(numSteps)
End Sub

Function GenerateSteps()
' Initiate the calibration and query the number of steps
scpi.Execute("sens:corr:coll:guid:init")
GenerateSteps = scpi.Execute("sens:corr:coll:guid:steps?")
End Function

Sub MeasureAndComplete(numSteps)
MsgBox("Number of steps is " + CStr(numSteps))
' Measure the standards
For i = 1 To numSteps
step = "Step " + CStr(i) + " of " + CStr(numSteps)
strPrompt = scpi.Execute("sens:corr:coll:guid:desc? " + CStr(i))
MsgBox strPrompt, vbOKOnly, step
' Note: if you have set up a slow sweep speed (for example, if
' you're using a narrow IF bandwidth) or you're using ECal, and
' while a cal step is being measured you wish to have your program
' perform other operations (like checking for the click event of a
' Cancel button) and you're NOT using the COM ScpiStringParser,
' you can use the optional ASYNchronous argument with the ACquire
' command as shown in this commented-out line below. The SCPI

```

```
' parser then will return immediately while the cal step measurement
' proceeds (i.e., the parser will NOT block-and-wait for the
' measurement step to finish, so you can send additional commands
' in the meantime). So you can do "*ESR?" or "*STB?" queries to
' monitor the status register bytes to see when the OPC bit gets set,
' which indicates the cal measurement step has finished. This OPC
' detection works for all of the PNA's SCPI parsers except the COM
' ScpiStringParser.
' "sens:corr:coll:guid:acq STAN" + CStr(i) + ",ASYN;*OPC"
scpi.Execute("sens:corr:coll:guid:acq STAN" + CStr(i))
Next
' Conclude the calibration
scpi.Execute("sens:corr:coll:guid:save")
MsgBox ("Cal is done!")
End Sub
```

Last modified:

5-Apr-2011 edited for ECal options

Perform a Simple Source Power Cal

This example performs a Source Power Cal using ONE USB Power Sensor, already connected to the PNA.

A measurement must first be set up with desired frequency range, power, and so forth, ready to be calibrated.

The SCPI commands in this example are sent over a COM interface using the [SCPIStringParser](#) object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the PNA hard drive as spc.vbs.

[Learn how to setup and run the macro.](#)

[See Source Power Cal SCPI commands](#)

See Other SCPI Example Programs

```
'Performs a source power cal on channel 1 - port 1 using a USB power sensor
'This example assumes ONE USB power sensor is connected to the PNA

Dim app

Dim scpi

Dim sensor

' Create / Get the PNA application.

Set app = CreateObject("AgilentPNA835x.Application")

Set scpi = app.ScpiStringParser

scpi.parse "SYST:PRES"

'set power accuracy tolerance and iterations

scpi.parse "SOUR1:POW1:CORR:COLL:ITER:NTOL 0.1"

scpi.parse "SOUR1:POW1:CORR:COLL:ITER:COUN 15"

'set power sensor settling tolerance

scpi.parse "SOUR1:POW1:CORR:COLL:AVER:NTOL 0.1"
```

```
scpi.parse "SOUR1::POW1:CORR:COLL:AVER:COUN 15"

'set offset value for amp or attenuation

scpi.parse "SOUR1:POW1:CORR:OFFS 0 DB"

'show source power cal dialog

scpi.parse "SOUR1:POW1:CORR:COLL:DISP ON"

'read the usb power sensor ID string

sensor=scpi.parse("SYST:COMM:USB:PMET:CAT?")

'specify that sensor

scpi.parse "SYST:COMM:PSEN usb," + sensor

'do the measurement

scpi.parse "SOUR1:POW1:CORR:COLL:ACQ PMR,"ASENSOR""

'save the source cal and create an R-Channel response calset

scpi.parse "SOUR:POW:CORR:COLL:SAVE RREC"
```

Last Modified:

16-Jul-2010	Fixed typo
26-Feb-2009	MX New topic

Perform a Source and Receiver Power Cal using SCPI

Programming the PNA using COM or using SICL/VISA over LAN (as in this example) leaves the PNA free to control GPIB devices as needed.

The first example, using Visual Basic, demonstrates the following:

- Performing a source power calibration of Port 2 for Channel 1.
- Reading the calibration data.

The second example performs a [Receiver Power Cal](#) using VBScript.

Learn more about [Power Calibrations](#).

See an example that [Uploads a Source Power Cal](#).

Other SCPI Example Programs

To run this program, you need:

- One of the following power meters connected to the PNA through GPIB: E4416A, E4417A, E4418A/B, E4419A/B, 437B, 438A, EPM-441A, EPM-442A

Note: If your power meter is other than these, you can [create your own Power Meter Driver](#) using our template.

- Your PC and PNA both connected to a LAN (for communicating with each other).
- The SICL and VISA components of Agilent's I/O Libraries software installed on your PC (both are included when you install the software, unless you already have another vendor's VISA installed. Then specify Full SICL and VISA installation to overwrite the other vendor's VISA).
- The module visa32.bas added to your VB project.
- A form with one button labeled cmdRun.
- A VISA interface configured on your remote PC to control the PNA. This could be GPIB interface or a [VISA LAN Client](#).
- On the PNA connect a Thru cable from port 1 to port 2.

Note: The [SOURCE:POWER:CORREction:COLLect:ACQuire](#) command, when used with a power

meter, cannot be sent over the GPIB unless the power meter is connected to a different GPIB interface. See the alternative methods described in the command details.

```
'Session to VISA Default Resource Manager

Private defRM As Long

'Session to PNA

Private viPNA As Long

'VISA function status return code

Private status As Long

Private Sub Form_Load()

defRM = 0

End Sub

Private Sub cmdRun_Click()

' String to receive data from the PNA.

' Dimensioned large enough to receive scalar comma-delimited values

' for 21 frequency points (20 ASCII characters per point)

Dim strReply As String * 420

Dim strStimulus, strCalValue

Dim strResult As String

' Open the VISA default resource manager

status = viOpenDefaultRM(defRM)

If (status < VI_SUCCESS) Then HandleVISAError

' Open a session (viPNA) to the PNA at "address 16" on the VISA

' interface configured as "GPIB1" on this PC. This could be a
```

```
' VISA LAN Client pointing to the SICL LAN Server on the PNA, or
' an actual GPIB interface on this PC connected to the PNA GPIB
' (in which case the power meter would need to be connected to a
' different GPIB interface on the PNA, such as the Agilent 82357A
' USB-to-GPIB).

status = viOpen(defRM, "GPIB0::16::INSTR", 0, 0, viPNA)

If (status < VI_SUCCESS) Then HandleVISAError

' Set the number of sweep points to 21 on Channel 1.

status = myGPIBWrite(viPNA, "SENS1:SWE:POIN 21")

If (status < VI_SUCCESS) Then HandleVISAError

' Specify the GPIB address of the power meter
' that will be used in performing the calibration.

status = myGPIBWrite(viPNA, "SYST:COMM:GPIB:PMET:ADDR 13")

If (status < VI_SUCCESS) Then HandleVISAError

' Turn use of the loss table OFF (this assumes there is
' virtually no loss in the RF path to the power sensor
' due to a splitter, coupler or adapter).

status = myGPIBWrite(viPNA, "SOUR:POW:CORR:COLL:TABL:LOSS OFF")

If (status < VI_SUCCESS) Then HandleVISAError

' Turn frequency checking OFF (so one power sensor is used for the entire cal
' acquisition sweep regardless of frequency span).

status = myGPIBWrite(viPNA, "SOUR:POW:CORR:COLL:FCH OFF")

If (status < VI_SUCCESS) Then HandleVISAError
```

```
' Specify a nominal power accuracy tolerance (NTOLerance) in dB for the
calibration,

' and the maximum number (COUNT) of iterations to adjust power at each point,
' attempting to achieve within tolerance of the desired power.  If at any
stimulus

' point the power fails to reach within the set tolerance of the desired power
' after the maximum number of iterations, the power at that point will be set to
the

' value determined by the last iteration (the Source Power Cal dialog box will
' indicate the FAIL, but we can still apply the cal if desired when it's
complete).

' Each iteration is based upon a SETTLED power reading (see comments preceding
the

' next two commands below).

status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:COLL:ITER:NTOL 0.1")

If (status < VI_SUCCESS) Then HandleVISAError

status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:COLL:ITER:COUN 3")

If (status < VI_SUCCESS) Then HandleVISAError

' The worst-case window of power uncertainty (for a calibration which meets
' tolerance) is the sum of the iteration tolerance and the power meter settling
' tolerance (which is described below).

' At each stimulus point, the PNA takes power meter readings and determine when
' they have settled by comparing the magnitude difference between consecutive
' readings versus a nominal dB tolerance limit (NTOLerance) on that magnitude
' difference.  When consecutive readings are within tolerance of each other, or
' if they are not within tolerance but we've taken a maximum number of readings
' (COUNT), the PNA does a weighted average of the readings taken at that stimulus
' point and that is considered our settled power reading.
```

```
status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:COLL:AVER:NTOL 0.1")

If (status < VI_SUCCESS) Then HandleVISAError

status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:COLL:AVER:COUN 5")

If (status < VI_SUCCESS) Then HandleVISAError

' Specify if the cal power level is offset (positive value for a gain, negative
' value for a loss) from the PNA port power setting on the channel when no source
' power cal is active. This is to account for components between the PNA test
' port and cal reference plane. In this example, we will calibrate at the PNA
' test port, so there is no offset (it is zero).

status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:OFFS 0 DB")

If (status < VI_SUCCESS) Then HandleVISAError

' Show the source power cal dialog during the source power cal acquisition.
' (this is the default, so this command is only necessary if this setting
' may have been changed beforehand, perhaps by another program).

status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:COLL:DISP ON")

If (status < VI_SUCCESS) Then HandleVISAError

' Clear the PNA's SCPI status registers because we are going to be monitoring
' the event status register to detect when the OPC bit gets set indicating
' the cal ACQUIRE completed.

status = myGPIBWrite(viPNA, "*CLS")

If (status < VI_SUCCESS) Then HandleVISAError

' Setting the I/O timeout value to 6000 milliseconds (6 seconds), because the
' PNA may take up to that amount of time to respond to some commands/queries
' while the cal ACQUIRE is progressing.

status = viSetAttribute(viPNA, VI_ATTR_TMO_VALUE, 6000)
```

```

If (status < VI_SUCCESS) Then HandleVISAError

' Specify the method (type of device) that will be used to perform the cal.
' Choose from power meter (PMETer), power meter and receiver (PMReceiver)
' or just receiver (RECeiver).

' PMReceiver uses the power meter for the first iteration of each point and
' the PNA's reference receiver for subsequent iterations, so is much faster
' than using power meter only. But the power meter accounts for compression
' when calibrating at the output of an active device, whereas the reference
' receiver cannot unless it is coupled to the cal reference plane (on a PNA
' which allows direct access to the receivers).

' Perform the source power cal acquisition sweep using the sensor attached to
' Channel A of the power meter (asking for an OPC reply when it's done). This
' assumes that the power sensor is already connected to Port 2 of the PNA.

' We'll put up an hourglass cursor while waiting for the acquire to complete.
Screen.MousePointer = vbHourglass

status = myGPIBwrite(viPNA, "SOUR1:POW:CORR:COLL:ACQ PMET,'ASEN','Port
2',ASYN;*OPC")

If (status < VI_SUCCESS) Then HandleVISAError

' Other valid selections would be the following:
' This mode uses Power Meter and Reference Receiver

'status = myGPIBwrite(viPNA, "SOUR1:POW:CORR:COLL:ACQ PMR,'BSEN','Port
2',ASYN;*OPC")

' This mode uses PNA receiver only (no power meter)

'status = myGPIBwrite(viPNA, "SOUR1:POW:CORR:COLL:ACQ REC,'b1','Port
2',ASYN;*OPC")

' Polling in a loop to detect when the OPC bit (bit 0, weight value 1) gets
' set in the Event Status Register indicating the ACquire finished. In this

```

```
' type of loop is where you could do other operations in-between the polling
' (like having your program's user-interface still respond to user input).
' If instead of Visual Basic you are programming in C or C++, as an
' alternative to having a polling loop like this, you could set up an SRQ
' handling function in your program (for example, see the documentation
' supplied with your vendor's implementation of VISA on how to register for
' callback when an SRQ event occurs).
```

```
Do
```

```
    status = myGPIBWrite(viPNA, "*ESR?")
```

```
    If (status < VI_SUCCESS) Then HandleVISAError
```

```
    status = myGPIBRead(viPNA, strReply)
```

```
    If (status < VI_SUCCESS) Then HandleVISAError
```

```
    esrByte = CByte(strReply)
```

```
Loop While (esrByte And 1) = 0
```

```
' Change mouse cursor from hourglass back to normal
```

```
Screen.MousePointer = vbDefault
```

```
' Conclude the calibration. This applies the cal data to PNA channel memory,
' and turns the correction ON for Port 2 on Channel 1,
' but does NOT save the calibration.
```

```
status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:COLL:SAVE")
```

```
If (status < VI_SUCCESS) Then HandleVISAError
```

```
' At this point, if you choose to save the instrument state as a ".CST" file,
' the calibration will be saved with the instrument state in that file.
```

```
' Prepare for doing data transfer in ASCII format.
```

```

status = myGPIBWrite(viPNA, "FORM:DATA ASCII")

If (status < VI_SUCCESS) Then HandleVISAError

' Read the stimulus values from Channel 1.

status = myGPIBWrite(viPNA, "SENS1:X?")

If (status < VI_SUCCESS) Then HandleVISAError

status = myGPIBRead(viPNA, strReply)

If (status < VI_SUCCESS) Then HandleVISAError

' Tokenize the reply string into an array containing the values

strStimulus = Split(strReply, ",")

' Read the source power correction data.

status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:DATA?")

If (status < VI_SUCCESS) Then HandleVISAError

status = myGPIBRead(viPNA, strReply)

If (status < VI_SUCCESS) Then HandleVISAError

' Tokenize the reply string into an array containing the values

strCalValue = Split(strReply, ",")

' Print the data using a message box (here, Chr returns the ASCII characters
' for Tab (9) and Linefeed (10)).

strResult = "Stimulus" & Chr(9) & Chr(9) & "Cal Value" & Chr(10)

For i = 0 To UBound(strStimulus)

strResult = strResult & Val(strStimulus(i)) & Chr(9) & Val(strCalValue(i)) &
Chr(10)

Next

```

```
MsgBox strResult

End Sub

Private Function myGPIBWrite(ByVal viHandle As Long, ByVal strOut As String) As Long

' The "+ Chr$(10)" appends an ASCII linefeed character to the
' output, for terminating the write transaction.

myGPIBWrite = viVPrintf(viHandle, strOut + Chr$(10), 0)

End Function

Private Function myGPIBRead(ByVal viHandle As Long, strIn As String) As Long
myGPIBRead = viVScanf(viHandle, "%t", strIn)
End Function

Sub HandleVISAError()

Dim strVisaErr As String * 200

Call viStatusDesc(defRM, status, strVisaErr)

MsgBox "*** Error : " + strVisaErr, vbExclamation

' Close the resource manager session (which also closes
' the session to the PNA).

If defRM <> 0 Then Call viClose(defRM)

End

End Sub

Public Sub Wait(ByVal mS_delay As Long)
```

```

Dim t0 As Single

t0 = Timer

Do While Timer - t0 < mS_delay / 1000

Dim dummy As Integer

dummy = DoEvents() ' if we cross midnight, back up one day

If Timer < t0 Then t0 = t0 - 86400

Loop

End Sub

```

Perform a Receiver Power Cal

The SCPI commands in this example are sent over a COM interface using the [SCPIStringParser](#) object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the PNA hard drive as *.vbs.

[Learn how to setup and run the macro.](#)

```

Dim pna
Dim scpi
Set pna = CreateObject("AgilentPNA835x.Application")
Set scpi = pna.ScpIStringParser
' For simplicity, this example starts from the preset instrument state
scpi.Execute "SYST:PRESet"
' Turn off continuous sweep
scpi.Execute "INITiate:CONTinuous OFF"
' Select the S11 measurement that was created by the instrument preset
scpi.Execute "CALCulate:PARAMeter:SElect 'CH1_S11_1'"
' Change the measurement parameter to measure the B receiver
scpi.Execute "CALCulate:PARAMeter:MODify B,1"
' Specify the Calibration Type, then Prompt
' to ensure the receiver is connected to port 1.
scpi.Execute "SENSe:CORRection:COLLect:METhod RPOWer"
MsgBox "Connect port 1 to port 2 so power is supplied to the B receiver, then press
enter"
' Acquire the power measurement; returning reply to *OPC? when finished.
response = scpi.Execute( "SENSe:CORRection:COLLect:ACQuire POWer;*OPC?" )
' Compute the error term, store to calset and turn on the calibration.
response = scpi.Execute( "SENSe:CORRection:COLLect:SAVE" )
MsgBox "Done with calibration."

```

Last modified:

9/22/06 Modified for receiver only feature

Perform a Source Power Cal with TWO Sensors

This example performs a Source Power Cal using TWO power sensors on a Dual-Channel Power Meter connected to the PNA via GPIB. Use of two power sensors is necessary when the frequency range of the measurement is greater than can be attained with a single sensor.

A measurement must first be set up with desired frequency range, power, and so forth, ready to be calibrated.

The SCPI commands in this example are sent over a COM interface using the [SCPIStringParser](#) object. You do not need to control the PNA via GPIB to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the PNA hard drive as spc.vbs.

[Learn how to setup and run the macro.](#)

[See Source Power Cal SCPI commands](#)

See Other SCPI Example Programs

```
'Performs a source power cal on channel 1 - port 1 using TWO USB power sensors.

Dim app

Dim scpi

Dim sensor

' Create / Get the PNA application.

Set app = CreateObject("AgilentPNA835x.Application")

Set scpi = app.ScpiStringParser

scpi.parse "SYST:PRES"

'set power accuracy tolerance and iterations

scpi.parse "SOUR1:POW1:CORR:COLL:ITER:NTOL 0.1"

scpi.parse "SOUR1:POW1:CORR:COLL:ITER:COUN 15"

'set power sensor settling tolerance

scpi.parse "SOUR1:POW1:CORR:COLL:AVER:NTOL 0.1"

scpi.parse "SOUR1:POW1:CORR:COLL:AVER:COUN 15"
```

```
'set offset value for amp or attenuation
scpi.parse "SOUR1:POW1:CORR:OFFS 0 DB"

'show source power cal dialog
scpi.parse "SOUR1:POW1:CORR:COLL:DISP ON"

'set frequency ranges for two power sensors
scpi.parse "SOUR:POW:CORR:COLL:ASEN 100E3, 3E9"
scpi.parse "SOUR:POW:CORR:COLL:BSEN 3E9,6E9"

'enable frequency check
scpi.parse "SOUR1:POW:CORR:COLL:FCHECK ON"

'specify the address of the power meter
scpi.parse "SYST:COMM:PSEN GPIB,'14'"

'do the measurements
MsgBox "Connect Sensor A to Port 1"

scpi.parse "SOUR1:POW1:CORR:COLL:ACQ PMR,'ASENSOR'"

MsgBox "Disconnect Sensor A from Port 1 and connect Sensor B to Port 1"

scpi.parse "SOUR1:POW1:CORR:COLL:ACQ PMR,'BSENSOR'"

'save the source cal and create an R-Channel response calset
scpi.parse "SOUR:POW:CORR:COLL:SAVE RREC"
```

Last Modified:

16-Jul-2010 MX New topic

Perform an ECal User Characterization

This example performs a user-characterization and stores it to both the ECal module memory and PNA disk memory. It also demonstrates the use of the EXPort, CLear, IMPort and 'KNAME:INF?' commands.

It then performs two 2-port calcs: the first using the characterization from module memory, then using the characterization from disk memory.

Note: This example requires that channel 1 be already calibrated.

The SCPI commands in this example are sent over a COM interface using the [SCPIStringParser](#) object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the PNA hard drive as ECal.vbs.

[Learn how to setup and run the macro.](#)

[See all ECal User Characterization SCPI commands](#)

See Other SCPI Example Programs

```
Option Explicit

Dim pna

Set pna = CreateObject("AgilentPNA835x.Application")

Dim scpi

Set scpi = pna.ScpiStringParser

' Substitute here the model number and serial number of your own ECal.
' Note that this example corresponds to a 4-port ECal module with
' serial number 00001. If you have a 2-port ECal module, their model
' numbers are '5x5' numbers -- for example, 'N4691-60001'.

Dim ecalModelNum

ecalModelNum = "N4433A"

Dim ecalSerialNum
```

```
ecalSerialNum = "00001"

scpi.Parse "SENS1:CORR:CKIT:ECAL:CHAR:ID '" & ecalModelNum & "'," & ecalSerialNum
& ""

MsgBox "ECal module to be characterized is: " &
scpi.Parse("SENS1:CORR:CKIT:ECAL:CHAR:ID?")

' Set which user characterization number (1-12) the new characterization
' will be stored to in the ECal module when it is done. If you intend to
' store your user characterization just to PNA Disk Memory and NOT the
' ECal module's memory, then omit this command.

Dim characterizationNumber
characterizationNumber = 1

scpi.Parse "SENS1:CORR:CKIT:ECAL:CHAR:CNUM " & CStr(characterizationNumber)

' The following commented-out lines of code show how you can access
' the list of connector type names you can set for the ports of an
' ECal when you user-characterize it. However, please note that if
' you are writing the user characterization to the ECal module's memory,
' as of yet only the Factory Defined set of connector choices will work
' properly (see SENS:CORR:CKIT:ECAL:CHAR:CONN:CAT?). If you will be saving
' your characterization to just PNA Disk Memory only, then all connector
' names returned by this query will work,
' user-defined connector names as well as factory-defined.

'Dim connTypeList
'connTypeList = scpi.Parse("SENS:CORR:CKIT:ECAL:CHAR:CONN:CAT?")
'MsgBox connTypeList

' For each port of the ECal module, specify which connector type
' is at the end of the adapter (or cable or fixture) that is
' connected to that port of the ECal for the characterization
```

```
' (must be one of the connector types that is included in the
' list that "SENS:CORR:CKIT:ECAL:CHAR:CONN:CAT?" returns). The
' default is "No adapter", which assumes you are characterizing that
' port of the ECal "as is" (nothing attached to it). So in this
' example, Ports C and D of the ECal are being characterized to just
' the ECal's connectors.

scpi.Parse "SENS1:CORR:CKIT:ECAL:CHAR:CONN:PORT1 'APC 3.5 male'" ' ECal Port A
scpi.Parse "SENS1:CORR:CKIT:ECAL:CHAR:CONN:PORT2 'APC 3.5 male'" ' ECal Port B

' As with the connector types, the information set in these next
' few properties also gets stored within the characterization.
' Set the name of the person and/or company that is producing
' this characterization.

scpi.Parse "SENS1:CORR:CKIT:ECAL:CHAR:DESC:USER 'John Doe, Acme Inc.'"
' Set user-specified description of the PNA being used.

scpi.Parse "SENS1:CORR:CKIT:ECAL:CHAR:DESC:VNA 'SN US12345678'"
' Set descriptions of what you have connected to the ECal module's
' ports for the characterization.
' Port A of the ECal

scpi.Parse "SENS1:CORR:CKIT:ECAL:CHAR:DESC:PORT1 '3.5 mm adapter, SN 00001'"
' Port B of the ECal

scpi.Parse "SENS1:CORR:CKIT:ECAL:CHAR:DESC:PORT2 '3.5 mm adapter, SN 00002'"

' Note that the "SENS:CORR:CKIT:ECAL:CHAR:" INITiate, ACQuire and SAVE
' ("CHAR:SAVE" but not "CHAR:DMEMemory:SAVE") commands can all each take a
significant
' amount of time to execute/complete. If you are looking at this example to
' leverage this functionality into a SCPI via GPIB or SCPI via SICT-LAN
' (VXI-11.2/11.3) application, then you could issue the "*CLS" and "*ESE 1"
commands
' as shown in the commented-out lines below, and use your I/O libraries' Serial
```

Poll

```
' function to repeatedly read the status byte until you detect bit 5 (weight of
32)

' in that byte is set. That will happen when the command you are pairing with

' ";*OPC" has completed its operation. But that technique only works for the
GPIB

' and SICL-LAN interfaces. If you need to use the TCPIP Socket or COM

' ScpiStringParser (as is used in this example) SCPI interfaces where there's

' no "built-in" Serial Poll type of function, to ensure your program operates in
a

' synchronized manner it will need to wait on the "*OPC?" reply (and not time
out)

' before proceeding to the next line of your program. In that event, we
recommend

' you execute these commands on a thread of execution separate from your
program's

' user interface thread.

' Of the "SENS:CORR:CKIT:ECAL:CHAR:" INITiate, ACQuire and SAVE commands, the
SAVE

' command takes the longest amount of time to complete (unless you've set up your

' measurement channel to have a very slow sweep time, in which case the ACQuire

' command could take longer). For an ECal that is a N469x, N4432A or N4433A, or
an

' 8509x or N4431x produced by Agilent in 2005 or later, the SAVE command can take
a

' maximum of approximately 4 to 5 minutes to complete (that corresponds to a

' characterization that will result in the ECal's memory becoming completely
filled).

' For an 8509x or N4431x ECal that was produced in 2004 or earlier, the SAVE
command

' can take a maximum of 9 to 10 minutes to complete (again that corresponds to a

' characterization that will result in the ECal's memory becoming completely
full).
```

```

' Begin a user characterization on Channel 1.

' If you will be storing this characterization to the ECal module's memory, then
' the boolean argument to this command is optional (but if you choose to include
it
' for that case then you must specify it as 1 or ON). If you will be storing
this
' characterization to PNA disk memory ONLY, then you should specify 0 or OFF for
' that argument. In this example we will be storing the characterization to both
' module memory and PNA disk memory, so we can just omit the argument and let it
' default to 1.

scpi.Parse "SENS1:CORR:CKIT:ECAL:CHAR:INIT"

Dim numSteps

numSteps = CLng( scpi.Parse("SENS1:CORR:CKIT:ECAL:CHAR:STEP?") )

Dim opcReply

'Dim statusByte

' Measure the steps.

' Note: prior to measuring the steps you must already have a calibration of the
' necessary number of ports applied to the channel (which in this example is
Channel 1).

' Otherwise an error will be reported to the SCPI error queue.

Dim i

For i = 1 To numSteps

    ' Display the step's description.

    MsgBox scpi.Parse("SENS1:CORR:CKIT:ECAL:CHAR:DESC? " & i)

    ' Clear the instrument's Status Byte.

    ' scpi.Execute "*CLS"

    ' Enable for the OPC bit (bit 0, which has weight 1) in the instrument's
    ' Event Status Register, so that when that bit's value transitions from 0 to 1
    ' then the Event Status Register bit in the Status Byte (bit 5 of that byte,

```

```

' weight 32) will become set.

' scpi.Execute "*ESE 1"

' Issue the ACQUIRE command

opcReply = scpi.Parse("SENS1:CORR:CKIT:ECAL:CHAR:ACQ STAN" & CStr(i) & ";*OPC?"
")

' scpi.Execute "SENS1:CORR:CKIT:ECAL:CHAR:ACQ STAN" & CStr(i) & ";*OPC"

' Do

' here is where if you leverage this example into an environment where
' you are using SCPI via GPIB or SICTL-LAN, that in this loop you could do a
' Serial Poll via that interface to read the status byte into this
' statusByte variable. Then this If statement would detect when bit 5 is
set.

' If ( (statusByte/32) Mod 2) Then Exit Do

' And note that normally you would want to have your program do some other
' processing (for example, check for user input from keyboard/mouse, for
' a cancellation request) here in this loop.

' Loop

MsgBox "ACQUIRE is complete"

Next

MsgBox "Now the user characterization will be saved to the ECal module and to PNA
disk memory"

'scpi.Execute "*CLS;*ESE 1"

' Save the user characterization to the ECal module's memory.

opcReply = scpi.Parse("SENS1:CORR:CKIT:ECAL:CHAR:SAVE;*OPC?")

'scpi.Execute "SENS1:CORR:CKIT:ECAL:CHAR:SAVE;*OPC"

'Do

' again here you could do a Serial Poll to get statusByte if using GPIB or
SICTL-LAN

' If ( (statusByte/32) Mod 2) Then Exit Do

'Loop

```

```

' Save the user characterization to PNA Disk Memory.

Dim characterizationName

characterizationName = "test"

opcReply = scpi.Parse("SENS1:CORR:CKIT:ECAL:CHAR:DMEM:SAVE '" &
characterizationName & "';*OPC?")

Dim pnaDiskMemCalKitName

pnaDiskMemCalKitName = GetCalKitName(characterizationName)

' Exporting the characterization from PNA disk memory into a file.

' The file can be used for loading the characterization into PNA disk memory on
another PNA.

scpi.Parse "SENS:CORR:CKIT:ECAL:EXP '" & pnaDiskMemCalKitName & "'"

' Demonstrating that the characterization can be cleared from PNA disk memory and
' then re-loaded (IMPorted) from the file that was created by the
' "SENS:CORR:CKIT:ECAL:EXP".

scpi.Parse "SENS:CORR:CKIT:ECAL:DMEM:CLE '" & pnaDiskMemCalKitName & "'"

scpi.Parse "SENS:CORR:CKIT:ECAL:DMEM:IMP 'C:/Program Files/Agilent/Network
Analyzer/ECal User Characterizations/" & pnaDiskMemCalKitName & ".euc'"

Dim moduleMemCalKitName

moduleMemCalKitName = GetCalKitName("User " & CStr(characterizationNumber))

MsgBox "Information about the characterization from ECal module memory = " &
scpi.Parse("SENS:CORR:CKIT:ECAL:KNAM:INF? '" & moduleMemCalKitName & "'")

MsgBox "Information about the characterization from PNA disk memory = " &
scpi.Parse("SENS:CORR:CKIT:ECAL:KNAM:INF? '" & pnaDiskMemCalKitName & "'")

MsgBox "User characterization is complete. Now we will calibrate using it.
First we will use it from ECal module memory."

DoTwoPortCal moduleMemCalKitName

MsgBox "Now we will calibrate using the characterization from PNA Disk Memory."

DoTwoPortCal pnaDiskMemCalKitName

MsgBox "Example has completed"

'

Function GetCalKitName(characterizationName)

```

```

Dim calKitName

calKitName = ecalModelNum

If Len(characterizationName) > 0 Then calKitName = calKitName & " " &
characterizationName

calKitName = calKitName & " ECal " & ecalSerialNum

GetCalKitName = calKitName

End Function

Sub DoTwoPortCal(calKitName)

' Specify the DUT connector for each PNA port to be calibrated (DUT connector =
ECal characterization's connector)

scpi.Parse "SENS1:CORR:COLL:GUID:CONN:PORT1 'APC 3.5 male'"
scpi.Parse "SENS1:CORR:COLL:GUID:CONN:PORT2 'APC 3.5 male'"

' Specify the "cal kit" for each of those ports
scpi.Parse "SENS1:CORR:COLL:GUID:CKIT:PORT1 '" & calKitName & "'"
scpi.Parse "SENS1:CORR:COLL:GUID:CKIT:PORT2 '" & calKitName & "'"

' This results in a calibration sequence of a single "connection step"
scpi.Parse "SENS1:CORR:COLL:GUID:INIT"

' Acquire the cal connection step

opcReply = scpi.Parse("SENS1:CORR:COLL:GUID:ACQ STAN1;*OPC?")

' Again here instead of waiting for opcReply you could do a Serial Poll to get
statusByte if using GPIB or SICTL-LAN

'scpi.Execute "SENS1:CORR:COLL:GUID:ACQ STAN1;*OPC"

'Do

' If ( (statusByte/32) Mod 2) Then Exit Do

'Loop

' Conclude the cal and turn it on

scpi.Parse "SENS1:CORR:COLL:GUID:SAVE"

End Sub

```

Last Modified:

18-Sep-2009	Updated with new User Char commands (9.0)
25-Nov-2008	New topic (8.33)

Perform an Unguided Cal on a 4-Port PNA

This topic describes how to perform an unguided calibration on a multiport network analyzer using SCPI. The objective here is to make clear the relationship between the physical port on which a standard is being measured, the actual device in the cal kit, and the SCPI command used to acquire the device.

There are two sets of SCPI commands that acquire calibrations. One set is used for guided cal, the other for unguided. The SCPI commands that provide remote access to unguided cal are in the SENS:CORR:COLL block:

- SENS:CORR:COLL:METHod
- SENS:CORR:COLL:ACQuire
- SENS:CORR:COLL:SAVE

On a four port network analyzer, the remote programmer needs to be aware of the relationship between the physical port and the calibration kit class assignments. The example program (below) illustrates the usage by performing three unique 2 port cals, taking care to acquire the appropriate standards.

Calibration standards classes are ‘categories’ of standard types. To perform a 2 port calibration, the cal wizard requires the user to measure:

3 reflection standards on the forward port:

- Class S11A typically an open
- Class S11B typically a short
- Class S11C typically a load

Likewise, 3 reflection standards are required for the reverse port:

- Class S22A typically an open
- Class S22B typically a short
- Class S22C typically a load

There is also a transmission standard that is measured in both directions:

- Class S21T typically a thru

The following illustrates the relationship between cal kit physical standards and calibration classes.

Here is a list of the physical devices in my calibration kit.

Standard #1 = "3.5 mm male short"

Standard #2 = "3.5 mm male open"

Standard #3 = "3.5 mm male broadband load"

Standard #4 = "Insertable thru standard"

Standard #5 = "3.5 mm male sliding load"

Standard #6 = "3.5 mm male lowband load"

Standard #7 = "3.5 mm female short"

Standard #8 = "female to female characterized thru adapter"

Standard #9 = "0-2 Load"

Standard #10 = "Open"

Standard #11 = "Non-insertable thru"

Standard #12 = "3.5 mm female lowband load"

Standard #13 = "3.5 mm female sliding load"

Standard #14 = "3.5 mm female broadband load"

Standard #15 = "3.5 mm female open"

When you perform a calibration remotely using SCPI, you don't specify the device number directly.

Rather, you specify the class you want to measure. Each device in the calibration kit is assigned to a class. And since more than one device can be assigned to the same class, each class contains an ordered list of devices. The class assignments are user-settable using the Advanced Modify Cal Kit dialog or the SCPI command:

[SENS:CORR:COLL:CKIT:ORDer](#)<class>, <std>, <std>, <std>, <std>,<std>,<std>,<std>

The 85052B kit used in the example program had the following standard list for each class: The list was obtained by issuing the corresponding SCPI query:

[SENS:CORR:COLL:CKIT:OLIST1?](#) S11A = +2,+15,+0,+0,+0,+0,+0

[SENS:CORR:COLL:CKIT:OLIST2?](#) S11B = +1,+7,+0,+0,+0,+0,+0

SENS:CORR:COLL:CKIT:OLIST3? S11C = +6,+5,+3,+12,+13,+14,+0

SENS:CORR:COLL:CKIT:OLIST4? S21T = +4,+8,+0,+0,+0,+0,+0

SENS:CORR:COLL:CKIT:OLIST5? S22A = +2,+15,+0,+0,+0,+0,+0

SENS:CORR:COLL:CKIT:OLIST6? S22B = +1,+7,+0,+0,+0,+0,+0

SENS:CORR:COLL:CKIT:OLIST7? S22C = +6,+5,+3,+12,+13,+14,+0

SENS:CORR:COLL:CKIT:OLIST8? S12T = +4,+8,+0,+0,+0,+0,+0

When you perform the calibration, you acquire data by issuing the ACQUIRE command:

[SENS:CORR:COLL:ACQ <class>\[, <subst> \]](#)

For example:

[SENS:CORR:COLL:SFOR 1](#)

SENS:CORR:COLL:ACQ STANA, SST2

The SFOR command tells the wizard to make the next acquisition in the forward direction. The ACQUIRE command specifies that we are measuring the 2nd device in the list for STANA. And since we are measuring SFORward, STANA refers to class #1 or S11A. The list of devices for this class are specified in the OLIST1 query above. The associations are shown in red.

Alternately, you could modify the device order for the S11A class to move device #15 into the first position (SENS:CORR:COLL:CKIT:ORDER1). When the desired device is in the first position, you needn't specify the order number in the ACQUIRE command. The default is the first device in the OLIST. This worked well for two port network analyzers where the order for S11A,B,C classes were setup for port 1 and the order for S22A,B,C was set up for port 2. With the kit setup in the proper order, you could eliminate the specification of the substandard number (SST<n>).

When performing 2 port calibrations on 4 Port Network Analyzers, the wizard applies S11A,B,C standards to the lower numbered port, S22A,B,C standards to the higher numbered port. Since the two classes (S11A,B,C and S22A,B,C) are applied to multiple ports, the programmer must take into account the ports being measured and take greater care when specifying the ACQUIRE command to ensure that the correct device is being measured.

Port to class relationship

Ports	S11A Port	S22A Port
1,2	1	2
1,3	1	3
1,4	1	4
2,3	2	3
2,4	2	4
3,4	3	4

The following example program shows one method of handling two port calcs on a multiport network analyzer. The connectors at the measurement plane are assumed to be (1) male, (2) female, (3) male, and (4) male. In the example, three calcs are performed: 1-2 (insertable male to female), 2-3 (insertable female to male), and 3-4 (noninsertable using an characterized adapter).

```
option explicit
public scpi
public pna
' assume a 4 port PNA with the following connectors:
' the standard measured on these ports will be the opposite gender
' PORT 1 = 3.5 male
' PORT 2 = 3.5 female
' PORT 3 = 3.5 male
' PORT 4 = 3.5 male
'To perform 2 port calibrations between 1-2, 2-3, and 3-4 you need to do the
following

call main

sub main
set pna = CreateObject("AgilentPnA835x.Application")
set scpi = pna.ScpiStringParser
pna.Preset
' select a kit to use for this demonstration
' kit #1 for the N5230A is the 85052B 3.5mm kit with sliding load
scpi.execute("SENS:CORR:COLL:CKIT:SELECT 1" )
PrintKitStandardInfo 1
PrintKitOlist 1

' -----
'   CALIBRATE PORTS 1 and 2, insertable cal
' -----

wscript.echo
wscript.echo "Calibrating ports 1 and 2"
scpi.execute("SYST:PRES;")
scpi.execute("calc:par:sel CH1_S11_1")
scpi.execute("SENS:CORR:TST:STATE 0")
```

```

scpi.execute("SENS:CORR:COLL:METHod SPARSOLT")
scpi.execute("SENS:CORR:SFOR 1")
MeasureFemaleStandards 1
scpi.execute("SENS:CORR:SFOR 0")
MeasureMaleStandards 2
MeasureTransmissionStandards 1,2
scpi.execute("SENS:CORR:COLL:SAVE")

' -----
'   CALIBRATE PORTS 2 and 3, insertable cal
' -----

wscript.echo
wscript.echo "Calibrating ports 2 and 3"
scpi.execute("SYST:PRES;")
scpi.execute("calc:par:sel CH1_S11_1")
scpi.execute("calc:par:mod S23")
scpi.execute("SENS:CORR:TST:STATE 0")
scpi.execute("SENS:CORR:COLL:METHod SPARSOLT")
scpi.execute("SENS:CORR:SFOR 1")
MeasureMaleStandards 2
scpi.execute("SENS:CORR:SFOR 0")
MeasureFemaleStandards 3
MeasureTransmissionStandards 2,3
scpi.execute("SENS:CORR:COLL:SAVE")

' -----
'   CALIBRATE PORTS 3 and 4, non-insertable cal
' -----

wscript.echo
wscript.echo "Calibrating ports 3 and 4"
scpi.execute("SYST:PRES;")
scpi.execute("calc:par:sel CH1_S11_1")
scpi.execute("calc:par:mod S43")
scpi.execute("SENS:CORR:COLL:METHod SPARSOLT")
scpi.execute("SENS:CORR:SFOR 1")
MeasureFemaleStandards 3
scpi.execute("SENS:CORR:SFOR 0")
MeasureFemaleStandards 4
MeasureAdapter 3, 4
scpi.execute("SENS:CORR:COLL:SAVE")
end sub

sub MeasureMaleStandards ( portNumber )
dim portstr
portstr = formatnumber(portNumber,0)
Promptconnect1 1, 1, portNumber
scpi.execute("SENS:CORR:COLL:ACQ STAN1;*OPC?")

Promptconnect1 2, 1, portNumber
scpi.execute("SENS:CORR:COLL:ACQ STAN2;*OPC?")
Promptconnect1 3, 3, portNumber
scpi.execute("SENS:CORR:COLL:ACQ STAN3,SST3;*OPC?")

```

```

end sub

sub MeasureFemaleStandards (portNumber)
dim portstr
portstr = formatnumber(portNumber,0)
Promptconnect1 1, 2, portNumber
scpi.execute("SENS:CORR:COLL:ACQ STAN1,SST2;*OPC?")
Promptconnect1 2, 2, portNumber
scpi.execute("SENS:CORR:COLL:ACQ STAN2,SST2;*OPC?")
Promptconnect1 3, 6, portNumber
scpi.execute("SENS:CORR:COLL:ACQ STAN3,SST6;*OPC?")
end sub

sub MeasureTransmissionStandards( port1, port2)
dim plstr
dim p2str
plstr = formatnumber( port1, 0)
p2str = formatnumber( port2, 0)

Promptconnect2 4, 1, port1, port2
scpi.execute("SENS:CORR:COLL:ACQ STAN4;*OPC?")
end sub

sub MeasureAdapter( port1, port2)
dim plstr
dim p2str
plstr = formatnumber( port1, 0)
p2str = formatnumber( port2, 0)

Promptconnect2 4, 2, port1, port2
scpi.execute("SENS:CORR:COLL:ACQ STAN4,SST2;*OPC?")
end sub

' return the nth item in the comma separated list
Function GetItemNumber( list, n)
dim strVector
strVector = split(list,",",-1,1)
GetItemNumber = strVector(n-1)
end function

' remove the trailing newline from str
function chop( str )
dim tmp
tmp = str
' remove the appended newline
dim pos
pos = InStrRev(tmp,vblf)
if (pos >0) then
tmp = mid(tmp,1,pos-1)
end if
chop = tmp
end function

```

```

'return the label for the nth standard assigned to the class described by
class_index.
' if class_index = 1, class is S11A (STAN1)
' if class_index = 2, class is S11B (STAN2), etc
function GetStandardLabel( class_index, nth)
dim olist
dim stdnum
dim resp
olist = scpi.execute("SENS:CORR:COLL:CKIT:OLIST" + formatnumber(class_index,0)+"?")
stdnum = GetItemNumber( olist, nth)
scpi.execute("SENS:CORR:COLL:CKIT:STAN " + formatnumber(stdnum,0))
resp = scpi.execute("SENS:CORR:COLL:CKIT:STAN:Label?")
GetStandardLabel = chop(resp)
end function

sub PromptConnect1( class_index, nth, port)
wscript.echo "CONNECT " + GetStandardLabel( class_index, nth) + " to port " +
formatnumber(port,0)
end sub

sub PromptConnect2( class_index, nth, port1, port2)
wscript.echo "CONNECT " + GetStandardLabel( class_index, nth) + " between ports " +
formatnumber(port1,0) + " and " + formatnumber(port2,0)
end sub

' Print the order of standards per class for this kit
sub PrintKitOlist( kit )
dim i
dim cmd
dim resp
wscript.echo
dim olistcmd
olistcmd = "SENS:CORR:COLL:CKIT:OLIST"
' list the sub standards for each of the following classes
' S11A, S11B, S11C, FWD TRANS, FWD ISOL, S22A, S22B, S22C, REV TRANS, REV ISOL
for i = 1 to 8
cmd = olistcmd + formatNumber(i,0) + "?"
resp = scpi.execute(cmd)
wscript.echo cmd + "= " + chop(resp)
next
end sub

sub PrintKitStandardInfo( kit )
wscript.echo scpi.execute("SENS:CORR:COLL:CKIT:NAME?")
dim i
for i = 1 to 30
dim slabel
dim snum
snum = formatNumber(i,0)
scpi.execute("SENS:CORR:COLL:CKIT:STAN " + snum)
slabel=scpi.execute("SENS:CORR:COLL:CKIT:STAN:Label?")
wscript.echo "Standard #"+snum+ " = " + chop(slabel)

```

```
next
end sub
```

The output from this program is as follows:

Microsoft (R) Windows Script Host Version 5.6

Copyright (C) Microsoft Corporation 1996-2001. All rights reserved.

"85052B 3.5 mm with sliding load"

Standard #1 = "3.5 mm male short"

Standard #2 = "3.5 mm male open"

Standard #3 = "3.5 mm male broadband load"

Standard #4 = "Insertable thru standard"

Standard #5 = "3.5 mm male sliding load"

Standard #6 = "3.5 mm male lowband load"

Standard #7 = "3.5 mm female short"

Standard #8 = "female to female characterized thru adapter"

Standard #9 = "0-2 Load"

Standard #10 = "Open"

Standard #11 = "Non-insertable thru"

Standard #12 = "3.5 mm female lowband load"

Standard #13 = "3.5 mm female sliding load"

Standard #14 = "3.5 mm female broadband load"

Standard #15 = "3.5 mm female open"

Standard #16 = "Open"

Standard #17 = "Open"

Standard #18 = "Open"

Standard #19 = "Open"

Standard #20 = "Open"

Standard #21 = "Open"

Standard #22 = "Open"

Standard #23 = "Open"

Standard #24 = "Open"

Standard #25 = "Open"

Standard #26 = "Open"

Standard #27 = "Open"

Standard #28 = "Open"

Standard #29 = "Open"

Standard #30 = "Open"

SENS:CORR:COLL:CKIT:OLIST1?= +2,+15,+0,+0,+0,+0,+0

SENS:CORR:COLL:CKIT:OLIST2?= +1,+7,+0,+0,+0,+0,+0

SENS:CORR:COLL:CKIT:OLIST3?= +6,+5,+3,+12,+13,+14,+0

SENS:CORR:COLL:CKIT:OLIST4?= +4,+8,+0,+0,+0,+0,+0

SENS:CORR:COLL:CKIT:OLIST5?= +2,+15,+0,+0,+0,+0,+0

SENS:CORR:COLL:CKIT:OLIST6?= +1,+7,+0,+0,+0,+0,+0

SENS:CORR:COLL:CKIT:OLIST7?= +6,+5,+3,+12,+13,+14,+0

SENS:CORR:COLL:CKIT:OLIST8?= +4,+8,+0,+0,+0,+0,+0

Calibrating ports 1 and 2

CONNECT "3.5 mm female open" to port 1

CONNECT "3.5 mm female short" to port 1

CONNECT "3.5 mm female broadband load" to port 1

CONNECT "3.5 mm male open" to port 2

CONNECT "3.5 mm male short" to port 2

CONNECT "3.5 mm male broadband load" to port 2

CONNECT "Insertable thru standard" between ports 1 and 2

Calibrating ports 2 and 3

CONNECT "3.5 mm male open" to port 2

CONNECT "3.5 mm male short" to port 2

CONNECT "3.5 mm male broadband load" to port 2

CONNECT "3.5 mm female open" to port 3

CONNECT "3.5 mm female short" to port 3

CONNECT "3.5 mm female broadband load" to port 3

CONNECT "Insertable thru standard" between ports 2 and 3

Calibrating ports 3 and 4

CONNECT "3.5 mm female open" to port 3

CONNECT "3.5 mm female short" to port 3

CONNECT "3.5 mm female broadband load" to port 3

CONNECT "3.5 mm female open" to port 4

CONNECT "3.5 mm female short" to port 4

CONNECT "3.5 mm female broadband load" to port 4

CONNECT "female to female characterized thru adapter" between ports 3 and 4

Perform Global Delta Match Cal

The following program performs a [Global Delta Match Calibration](#). This may be required when performing an Unknown Thru Cal or TRL Cal on PNA-L models. [See example of Unknown Thru or TRL Cal.](#)

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Delta.vbs. [Learn how to setup and run the macro.](#)

```
Sub PerformGlobalDeltaMatchCal()
Set pna = CreateObject("AgilentPNA835x.Application")
Set scpi = pna.ScpiStringParser

' Initiate a Global Delta Match calibration, choosing connector and cal kit
scpi.Parse "SENS:CORR:COLL:GUID:DMAT 'APC 3.5 female', '85033D/E'"

' Query the number of calibration steps
retStr = scpi.Parse("SENS:CORR:COLL:GUID:STEP?")
numSteps = CInt(retStr)

' Measure the cal standards
For i = 1 To numSteps
  prompt = scpi.Parse("SENS:CORR:COLL:GUID:DESC? " & CStr(i))
  retVal = MsgBox(prompt, vbOKCancel)
  If retVal = vbCancel Then Exit Sub
  retStr = scpi.Parse("SENS:CORR:COLL:GUID:ACQ STAN" & CStr(i) & ";*OPC?")
Next

' Compute the error coefficients and save the cal to Global Delta Match CalSet
scpi.Parse "SENS:CORR:COLL:GUID:SAVE"
MsgBox "Cal is done!"
End Sub
```

Perform a Guided 1-Port Cal on Port 2

This VBScript program does the following:

1. Clear measurements from the PNA
2. Create a new S22 measurement
3. Set an instrument state
4. Select the connector types
5. Select a cal kit
6. Initiate a Guided calibration
7. Display a prompt to connect each standard
8. Save the calibration to a newly created cal set

Note: This example illustrates an important step when calibrating a reflection measurement in the reverse direction. You **MUST** create a reverse (S22) measurement and have it be the active (selected) measurement on the channel that is being calibrated. This is not necessary for any calibrating any other measurement parameter.

The SCPI commands in this example are sent over a COM interface using the [SCPIStringParser](#) object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Guided.vbs. [Learn how to setup and run the macro.](#)

```
Dim App
Set App = CreateObject("AgilentPNA835x.Application")
App.Preset

Dim step
Dim Parser
Dim prompt
Dim txtDat
Dim Chan

Rem Clear old measurements
App.Reset

Rem Create a new Measurement
Set Parser = App.SCPIStringParser
Parser.Parse "DISPlay:WINDow1:STATE ON"
```

```

Parser.Parse "CALCulate:PARAMeter:DEFine:EXT 'MyMeas',S22"
Parser.Parse "DISPlay:WINDow1:TRACel:FEED 'MyMeas'"

Rem Initialize state
Set Chan = App.ActiveChannel
Chan.StartFrequency = 200e6
Chan.StopFrequency = 1.5e9
Chan.IFBandwidth = 1000
step = 3

Rem Begin a guided calibration
Parser.Parse "SENS:CORR:COLL:GUID:CONN:PORT1 'Not used'"
Parser.Parse "SENS:CORR:COLL:GUID:CONN:PORT2 'Type N (50) male'"
Parser.Parse "SENS:CORR:COLL:GUID:CKIT:PORT1 ''"
Parser.Parse "SENS:CORR:COLL:GUID:CKIT:PORT2 '85054D'"
Parser.Parse "SENS:CORR:COLL:GUID:INIT"

Rem Query the number of steps
txtDat = Parser.Parse("SENS:CORR:COLL:GUID:STEP?")

Rem Display the number of steps
MsgBox("Number of steps is " + txtDat)

Rem Set the loop counter limit
step = txtDat

Rem Measure the standards
For i = 1 To step
If i= 1 Then
prompt = Parser.Parse("sens:corr:coll:guid:desc? 1")
MsgBox(prompt)
Parser.Parse ("sens:corr:coll:guid:acq STAN1")
ElseIf i = 2 then
prompt = Parser.Parse("sens:corr:coll:guid:desc? 2")
MsgBox(prompt)
Parser.Parse ("sens:corr:coll:guid:acq STAN2")
ElseIf i = 3 then
prompt = Parser.Parse("sens:corr:coll:guid:desc? 3")
MsgBox(prompt)
Parser.Parse ("sens:corr:coll:guid:acq STAN3")
End If
Next

Rem All standards have been measured. Save the result
Parser.Parse "SENS:CORR:COLL:GUID:SAVE"
MsgBox("The calibration has been completed")

```

Perform a Guided Calibration using SCPI

This VBScript program performs a Guided Calibration using ECal or Mechanical standards. This example includes optional ECal orientation features.

This example has been updated to include:

- Guided Power Cal (Oct 8, 2010)
- The setting of Unknown Thru or Adapter Removal adapter delay. (March 2006).
- The activation of a channel to be calibrated. (Aug. 2006).

The SCPI commands in this example are sent over a COM interface using the [SCPIStringParser](#) object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Guided.vbs. [Learn how to setup and run the macro.](#)

```
' Performing a Guided 2-port cal (Ports 1 and 2)
TwoPortGuidedCal
Sub TwoPortGuidedCal
Dim app
Dim scpi
Dim connList
Dim selectedConn1, selectedConn2
Dim kitList
Dim selectedKit
Dim message
' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

'The following demonstrates that the Active Channel is cal'd
```

```

'Preset the PNA

scpi.Execute "SYST:UPR"

'Create a new measurement on Chan 2

'Now there are two windows, channels and measurements

'This becomes the Active Measurement

scpi.Execute ("DISPlay:WINDow2:STATE ON")

'Define a measurement name, parameter

scpi.Execute ("CALCulate2:PARAMeter:DEFine:EXT 'MyMeas',S21")

'"FEED" the measurement

scpi.Execute ("DISPlay:WINDow2:TRACe1:FEED 'MyMeas'")

'This is the Active Measurement

'Activate the 'Preset' measurement to cal chan 1

scpi.Execute ("CALC1:PAR:SEL 'CH1_S11_1'")

' Query the list of connectors that the PNA system recognizes

connList = scpi.Execute("sens:corr:coll:guid:conn:cat?")

' Format the list with linefeed characters in place of the commas

connList = FormatList(connList)

message = "Enter your DUT connector for Port 1. Choose from this list:"

message = message & Chr(10) & Chr(10) & connList

' Select the connector for Port 1

selectedConn1 = InputBox(message)

If selectedConn1 = "" Then Exit Sub

scpi.Execute "sens:corr:coll:guid:conn:port1 '" & selectedConn1 & ""

message = "Enter your DUT connector for Port 2. Again, choose from this list:"

message = message & Chr(10) & Chr(10) & connList

' Select the connector for Port 2

selectedConn2 = InputBox(message)

If selectedConn2 = "" Then Exit Sub

```

```
scpi.Execute "sens:corr:coll:guid:conn:port2 '" & selectedConn2 & "'"  
  
' Note: If your PNA has more than 2 ports, then uncomment  
' one or both of these next two lines.  
  
'scpi.Execute "sens:corr:coll:guid:conn:port3 ""Not used"" "  
'scpi.Execute "sens:corr:coll:guid:conn:port4 ""Not used"" "  
  
' This next block of commented code demonstrates how to specify an adapter  
' and it's electrical delay, in situations where you are performing an  
' Unknown Thru or Adapter Removal calibration. In most situations, the  
' PNA is able to correctly determine an adapter's electrical length  
' at the end of the calibration. However, there are scenarios where  
' the PNA cannot correctly calculate the length -- such as when the channel  
' has a relatively small number of measurement points (for example, 201 or less)  
' and the adapter is significantly long (for example, a cable that is several  
feet).  
  
' In these cases, the ADAP commands (below) enable you to explicitly specify  
' the adapter you are using.  
  
' Send these commands prior to the "sens:corr:coll:guid:init" command.  
  
' Create adapter and return the adapter number  
  
'adapterNum = scpi.Execute("sens:corr:coll:guid:adap:cre? '" & selectedConn1 &  
'','"& selectedConn2 & "'")  
  
' The adapterNum string contains a '+' character.  
  
' Here we convert to integer to remove that.  
  
'adapterNum = CStr( CInt(adapterNum) )  
  
' Specify that this adapter has 10 nanoseconds electrical delay (coaxial).  
  
'scpi.Execute "sens:corr:coll:guid:adap" & adapterNum & ":del 10E-9"  
  
' Text description of adapter  
  
'scpi.Execute "sens:corr:coll:guid:adap" & adapterNum & ":desc 'My adapter'"  
  
' Select to use this adapter specifically between ports 1 and 2
```

```

'scpi.Execute "sens:corr:coll:guid:adap" & adapterNum & ":path 1,2"
' End of adapter block

' Query the list of acceptable cal kits and
' ECal module characterizations for Port 1.
kitList = scpi.Execute("sens:corr:coll:guid:ckit:cat? '" & selectedConn1 & "'")
' Format the list with linefeed
' characters in place of the commas
kitList = FormatList(kitList)
message = "Enter your cal kit or ECal module characterization for Port 1. "
message = message & "Choose from this list:"
message = message & Chr(10) & Chr(10) & kitList
' Select the Cal Kit or ECal module
' characterization to use for Port 1.
selectedKit = InputBox(message)
If selectedKit = "" Then Exit Sub
scpi.Execute "sens:corr:coll:guid:ckit:port1 '" & selectedKit & "'")
' Query the list of acceptable cal kits
' and ECal module characterizations for Port 2.
kitList = scpi.Execute("sens:corr:coll:guid:ckit:cat? '" & selectedConn2 & "'")
' Format the list with linefeed characters in place of the commas
kitList = FormatList(kitList)
message = "Enter your cal kit or ECal module characterization for Port 2. "
message = message & "Choose from this list:"
message = message & Chr(10) & Chr(10) & kitList
' Select the Cal Kit or ECal module
' characterization to use for Port 2.
selectedKit = InputBox(message)

```

```

If selectedKit = "" Then Exit Sub

scpi.Execute "sens:corr:coll:guid:ckit:port2 '" & selectedKit & "'"

' This determines whether the cal will be a "Guided Power Cal"
' or just a traditional S-parameter cal.

message = "On which port number shall power be measured?  "
message = message & "For a traditional guided cal without power cal, enter 0"

Dim powerPort

powerPort = CInt( InputBox(message) )

If powerPort > 0 Then

scpi.Execute("sens:corr:coll:guid:psen" & CStr(powerPort) & ":stat on")

Dim retVal

retVal = MsgBox("Is the power sensor's connector type or gender different from
the DUT connector for that port?", vbYesNo)

If retVal = vbYes Then

message = "Enter your power sensor's connector.  Choose from this list:"
message = message & Chr(10) & Chr(10) & connList

' Select the sensor's connector.

selectedConn1 = InputBox(message)

If selectedConn1 = "" Then Exit Sub

scpi.Execute "sens:corr:coll:guid:psen" & CStr(powerPort) & ":conn '" &
selectedConn1 & "'"

' Query the list of acceptable cal kits and ECal module characterizations
' that are applicable for the sensor's connector.

kitList = scpi.Execute("sens:corr:coll:guid:ckit:cat? '" & selectedConn1 & "'")

' Format the list with linefeed
' characters in place of the commas

kitList = FormatList(kitList)

message = "Enter your cal kit or ECal module characterization to use for de-embed
of the sensor's connector.  "

```

```

message = message & "Choose from this list:"

message = message & Chr(10) & Chr(10) & kitList

' Select the Cal Kit or ECal module characterization to use for de-embed of the
sensor's connector.

selectedKit = InputBox(message)

If selectedKit = "" Then Exit Sub

scpi.Execute "sens:corr:coll:guid:psen" & CStr(powerPort) & ":ckit '" &
selectedKit & "'"

Else

scpi.Execute("sens:corr:coll:guid:psen" & CStr(powerPort) & ":conn 'Ignored'")

End If ' End of block that considers the sensor's connector

' Ask for the power level to perform the power cal at
' (if this command is omitted, the default is 0 dBm).

Dim powerLevel

powerLevel = InputBox("Enter the power level for the power cal to be performed
at")

If powerLevel = "" Then Exit Sub

scpi.Execute "sens:corr:coll:guid:psen" & CStr(powerPort) & ":pow:lev " &
powerLevel

Else

scpi.Execute("sens:corr:coll:guid:psen1:stat off")

End If ' End of block that considers if the cal will include power calibration

' This next block of commented code
' shows optional functions when using ECal.
' Send these "sens:corr:pref" commands prior to the
' "sens:corr:coll:guid:init" command.
' Read ECAL information from ECal module #1 on the USB bus
' about the Agilent factory characterization data
'module1Info = scpi.Execute("sens:corr:coll:ckit:inf? ECAL1,CHAR0")

```

```

'MsgBox "Description of ECal Module #1:" & Chr(10) & Chr(10) & module1Info

' The following command enables auto orientation of
' the ECal module (The PNA senses which port of the
' module is connected to which port of the PNA).
'scpi.Execute "sens:corr:pref:ecal:ori ON"

' However, if you are measuring at very low power levels where
' the PNA may fail to sense the module's orientation, then turn auto
' orientation OFF and specify how the module is connected.
' "A1,B2" indicates Port A of the module is connected
' to PNA Port 1 and Port B is connected to PNA Port 2).
'scpi.Execute "sens:corr:pref:ecal:ori OFF"
'scpi.Execute "sens:corr:pref:ecal:pmap ECAL1,'A1,B2'"
' End of optional ECal setup

' Select the thru method of "Default". This instructs the PNA to
' determine which thru standard measurement technique to use
' based upon the selected connectors and
' calibration kit(s) and the PNA model number.
' with new CMET and TMET 'default' is set by not sending the commands
'
' Initiate the calibration and query the number of steps
scpi.Execute "sens:corr:coll:guid:init"

numSteps = scpi.Execute("sens:corr:coll:guid:steps?")

MsgBox "Number of steps is " + CStr(numSteps)

' Measure the standards

For i = 1 To numSteps

step = "Step " + CStr(i) + " of " + CStr(numSteps)

```

```

strPrompt = scpi.Execute("sens:corr:coll:guid:desc? " + CStr(i))
MsgBox strPrompt, vbOKOnly, step
scpi.Execute "sens:corr:coll:guid:acq STAN" + CStr(i)
Next
' Conclude the calibration
scpi.Execute "sens:corr:coll:guid:save"
MsgBox "Cal is done!"
End Sub
Function FormatList(list)
Dim tokens
' Strip the leading and trailing quotation
' marks from the list string
list = Mid(list, 2, Len(list) - 3)
' Tokenize the comma-delimited list string
' into an array of the individual substrings
tokens = Split(list, ",")
' Rebuild the list string, placing linefeed
' characters where the commas were,
' using Trim to remove leading and trailing spaces.
list = ""
For i = 0 To UBound(tokens)
tokens(i) = Trim(tokens(i))
list = list & tokens(i) & Chr(9)
If i < UBound(tokens) Then
i = i + 1
tokens(i) = Trim(tokens(i))
list = list & tokens(i) & Chr(10)

```

```
End If  
Next  
FormatList = list  
End Function
```

Last Modified:

8-Oct-2010 Updated for Enhanced Power (BH)
14-May-2007 MX Updated for new CMET and TMET commands

Perform Guided ECal using SCPI

This VBScript program performs a Guided ECal Calibration. While this example is good to use as a starting point for Guided ECal, the [Guided comprehensive cal example](#) has some advanced features that are not in this program.

The SCPI commands in this example are sent over a COM interface using the [SCPIStringParser](#) object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Guided.vbs. [Learn how to setup and run the macro.](#)

```
' Performing a 2-port cal (Ports 1 and 2)
Dim app
Dim scpi

' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

' Specify the DUT connectors
' (for each connector of your DUT, one of the ECal module's ports must have
' that same connector, or else you cannot achieve the cal using that module).
scpi.Execute "sens:corr:coll:guid:conn:port1 ""APC 3.5 female"" "
scpi.Execute "sens:corr:coll:guid:conn:port2 ""APC 3.5 male"" "

' Note: If your PNA has more than 2 ports, you would need to uncomment
' one or both of these next two lines, to explicitly specify this is
' just a 2-port cal.
'scpi.Execute "sens:corr:coll:guid:conn:port3 ""Not used"" "
'scpi.Execute "sens:corr:coll:guid:conn:port4 ""Not used"" "
MsgBox "Connectors defined for Ports 1 and 2"

' Specify ECal modules

scpi.Parse "sens:corr:coll:guid:ckit:port1 'N4691-60004 ECal'"

scpi.Parse "sens:corr:coll:guid:ckit:port2 'N4691-60004 ECal'"

' Non-factory characterizations are specified as follows:

'scpi.Parse "sens:corr:coll:guid:ckit:port2 'N4691-60004 User 1 ECal'"

' When two or more ECal modules with the same model number are connected
' also specify the serial number as follows:

'scpi.Parse "sens:corr:coll:guid:ckit:port2 'N4691-60004 ECal 01234'"
```

```
' When Disk Memory ECal user characterizations are used,
' specify both the User char and the serial number as follows:
'scpi.Parse "sens:corr:coll:guid:ckit:port2 'N4691-60004 MyDskChar ECal 01234'"
'
MsgBox "Cal kits defined for Ports 1 and 2"

' Initiate the calibration and query the number of steps
scpi.Execute "sens:corr:coll:guid:init"
numSteps = scpi.Execute("sens:corr:coll:guid:steps?")
MsgBox "Number of steps is " + CStr(numSteps)

' Measure the standards
For i = 1 To numSteps
step = "Step " + CStr(i) + " of " + CStr(numSteps)
strPrompt = scpi.Execute("sens:corr:coll:guid:desc? " + CStr(i))
MsgBox strPrompt, vbOKOnly, step
scpi.Execute "sens:corr:coll:guid:acq STAN" + CStr(i)
Next

' Conclude the calibration
scpi.Execute "sens:corr:coll:guid:save"
MsgBox "Cal is done!"
```

Last modified:

5-Apr-2011 edited for ECal options

Perform Guided Mechanical Cal using SCPI

This VBScript program performs a Guided Calibration using Mechanical standards. While this example is good to use as a starting point for guided mechanical cal, the [Guided comprehensive cal example](#) has some advanced features that are not in this program.

The SCPI commands in this example are sent over a COM interface using the [SCPIStringParser](#) object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Guided.vbs. [Learn how to setup and run the macro.](#)

' Performing a 2-port cal (Ports 1 and 2)

```
Dim app
Dim scpi

' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

' Specify the DUT connectors
scpi.Execute "sens:corr:coll:guid:conn:port1 ""APC 3.5 female"" "
scpi.Execute "sens:corr:coll:guid:conn:port2 ""APC 3.5 male"" "

' Note: If your PNA has more than 2 ports, you would need to uncomment
' one or both of these next two lines, to explicitly specify this is
' just a 2-port cal.
'scpi.Execute "sens:corr:coll:guid:conn:port3 ""Not used"" "
'scpi.Execute "sens:corr:coll:guid:conn:port4 ""Not used"" "
MsgBox "Connectors defined for Ports 1 and 2"

' Select the Cal Kit for each port being calibrated.
scpi.Execute "sens:corr:coll:guid:ckit:port1 ""85052D"" "
scpi.Execute "sens:corr:coll:guid:ckit:port2 ""85052D"" "
MsgBox "Cal kits defined for Ports 1 and 2"

' Initiate the calibration and query the number of steps
scpi.Execute "sens:corr:coll:guid:init"
numSteps = scpi.Execute("sens:corr:coll:guid:steps?")
MsgBox "Number of steps is " + CStr(numSteps)

' Measure the standards
'The following series of commands shows that standards
'can be measured in any order. These steps acquire
'measurement of standards in reverse order.
'It is easiest to iterate through standards using
'a For-Next Loop.
```

```
For i = numSteps To 1
step = "Step " + CStr(i) + " of " + CStr(numSteps)
strPrompt = scpi.Execute("sens:corr:coll:guid:desc? " + CStr(i))
MsgBox strPrompt, vbOKOnly, step
scpi.Execute "sens:corr:coll:guid:acq STAN" + CStr(i)
Next

' Conclude the calibration
scpi.Execute "sens:corr:coll:guid:save"
MsgBox "Cal is done!"
```

Last Modified:

20-Jan-2007 Added note about any order for steps.

Perform Guided TRL Calibration

This VBScript file performs a 2-Port Guided TRL calibration on **2-port PNA analyzers**. ([See an example of TRL cal on a 4-port PNA.](#)) This program does the following:

- Clear old measurements from the PNA
- Create a new S22 measurement
- Set an instrument state
- Select the connectors and cal kit
- Initiate a Guided calibration
- Display a prompt as each new standard must be connected
- Save the calibration to a newly created cal set.

Note: This program runs without error on all PNA code revisions 7.21 and higher.

The SCPI commands in this example are sent over a COM interface using the [SCPIStringParser](#) object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as TRL.vbs. [Learn how to setup and run the macro.](#)

```
Dim App
Dim Parser
Dim Chan
Dim txtDat
Dim step
Dim parserTxt
Dim prompt
Set App = CreateObject("AgilentPNA835x.Application")
' Clear old measurements
App.Reset
' Create a new Measurement
```

```
Set Parser = App.SCPIStrParser
Parser.Parse "DISPlay:WINDow1:STATE ON"
Parser.Parse "CALCulate:PARAmeter:DEFine:EXT 'MyMeas',S12"
Parser.Parse "DISPlay:WINDow1:TRACe1:FEED 'MyMeas'"
' Initialize state
Set Chan = App.ActiveChannel
Chan.StartFrequency = 18.0e9
Chan.StopFrequency = 20.0e9
Chan.IFBandwidth = 1000
' Begin a guided calibrations
Parser.Parse "SENS:CORR:COLL:GUID:CONN:PORT1 'APC 3.5 male'"
Parser.Parse "SENS:CORR:COLL:GUID:CONN:PORT2 'APC 3.5 female'"
Parser.Parse "SENS:CORR:COLL:GUID:CKIT:PORT1 '85052C'"
Parser.Parse "SENS:CORR:COLL:GUID:CKIT:PORT2 '85052C'"
' Select TRL cal method.
Parser.Parse "SENS:CORR:COLL:GUID:PATH:CMET 1,2,'TRL'"
txtDat = Parser.Parse("SENS:CORR:COLL:GUID:PATH:CMET? 1,2")
MsgBox("Method " + txtDat)
Parser.Parse "SENS:CORR:COLL:GUID:INIT"
' Query the number of steps
txtDat = Parser.Parse("SENS:CORR:COLL:GUID:STEP?")
' Display the number of steps
MsgBox("Number of steps is " + txtDat)
' Set the loop counter limit
step = CInt(txtDat)
' Measure the standards
```

```
For i = 1 To step
parserTxt = "sens:corr:coll:guid:desc? " + CStr(i)
prompt = Parser.Parse(parserTxt)
MsgBox(prompt)
parserTxt = "sens:corr:coll:guid:acq STAN" + CStr(i)
Parser.Parse (parserTxt)
Next
' All standards have been measured.  Save the result
Parser.Parse "SENS:CORR:COLL:GUID:SAVE"
MsgBox("The TRL calibration has been completed")
```

Last Modified:

9-Apr-2007 MX Updated for new CMethod command

Perform an Unguided 1-Port Cal on Port 2

This VBScript program does the following:

1. Clear measurements from the PNA
2. Create a new S22 measurement
3. Set an instrument state
4. Select a cal kit
5. Initiate an Unguided calibration
6. Display a prompt to connect each standard
7. Save the calibration to a newly created cal set

Note: This example illustrates an important step when calibrating a reflection measurement in the reverse direction. You **MUST** create a reverse (S22) measurement and have it be the active (selected) measurement on the channel that is being calibrated. This is not necessary for any calibrating any other measurement parameter.

The SCPI commands in this example are sent over a COM interface using the [SCPIStringParser](#) object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Unguided.vbs. [Learn how to setup and run the macro.](#)

```
Dim App
Set App = CreateObject("AgilentPNA835x.Application")
App.Preset

Dim Parser
Dim Chan

Rem Clear old measurements
App.Reset

Rem Create a new Measurement
Set Parser = App.SCPIStringParser
Parser.Parse "DISPlay:WINDow1:STATE ON"
Parser.Parse "CALCulate:PARAMeter:DEFine:EXT 'MyMeas',S22"
Parser.Parse "DISPlay:WINDow1:TRACe1:FEED 'MyMeas'"

Rem Initialize state
Set Chan = App.ActiveChannel
```

```
Chan.StartFrequency = 200e6
Chan.StopFrequency = 1.5e9
Chan.IFBandwidth = 1000

Rem Begin an unguided calibration
Rem Set the calibration method
Parser.Parse "SENSe:CORRection:COLLect:METhod REFL3"

Rem Turn off continuous sweep
Parser.Parse "INITiate:CONTinuous OFF"

Rem Select a cal kit
Parser.Parse "SENSe:CORRection:COLLect:CKIT:SElect 1"

Rem Measure the standards
MsgBox("Connect OPEN to port 2. Then press OK")
Parser.Parse ("sens:corr:coll:acq STAN1")

MsgBox("Connect SHORT to port 2. Then press OK")
Parser.Parse ("sens:corr:coll:acq STAN2")

MsgBox("Connect LOAD to port 2. Then press OK")
Parser.Parse ("sens:corr:coll:acq STAN3")

Rem All standards have been measured. Save the result
Parser.Parse "SENS:CORR:COLL:SAVE"

Rem Turn ON continuous sweep
Parser.Parse "INITiate:CONTinuous ON"
MsgBox("The calibration has been completed")
```

Perform an Unguided 2-Port Mechanical Cal

This VBScript program performs an Unguided, Full 2-Port, calibration using ONE set of mechanical calibration standards.

The SCPI commands in this example are sent over a COM interface using the [SCPIStringParser](#) object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Unguided.vbs. [Learn how to setup and run the macro.](#)

```
Set App = CreateObject("AgilentPNA835x.Application")
Set Scpi = App.SCPIStringParser

'Initialize state
Scpi.Execute ("SYSTem:PRESet")

'Select the Preset measurement
Scpi.Execute ("CALCulate:PARAMeter:SElect 'CH1_S11_1'")

'Set the calibration method
Scpi.Execute ("SENSe:CORRection:COLLect:METHOD SPARSOLT")

'Select a cal kit
Scpi.Execute ("SENSe:CORRection:COLLect:CKIT:SElect 1")

'Set one set of standards
Scpi.Execute ("SENSe:CORRection:TStandards OFF")

'Set acquisition to FORWARD
Scpi.Execute ("SENSe:CORRection:SFORward ON")

'Measure the standards in forward direction
MsgBox "Connect OPEN to Port 1; then press OK"
Scpi.Execute ("SENSe:CORRection:COLLect:ACQuire stan1")

MsgBox "Connect SHORT to Port 1; then press OK"
Scpi.Execute ("SENSe:CORRection:COLLect:ACQuire stan2")

MsgBox "Connect LOAD to Port 1; then press OK"
Scpi.Execute ("SENSe:CORRection:COLLect:ACQuire stan3")

'Set acquisition to REVERSE
Scpi.Execute ("SENSe:CORRection:SFORward OFF")

'Measure the standards in reverse direction
MsgBox "Connect OPEN to Port 2; then press OK"
Scpi.Execute ("SENSe:CORRection:COLLect:ACQuire stan1")
```

```
MsgBox "Connect SHORT to Port 2; then press OK"  
Scpi.Execute ("SENSe:CORRection:COLLect:ACQuire stan2")
```

```
MsgBox "Connect LOAD to Port 2; then press OK"  
Scpi.Execute ("SENSe:CORRection:COLLect:ACQuire stan3")
```

```
'Measure the thru standard
```

```
MsgBox "Connect THRU between Ports 1 and 2; then press OK"  
Scpi.Execute ("SENSe:CORRection:COLLect:ACQuire stan4")
```

```
'OPTIONAL Measure Isolation
```

```
MsgBox "Connect LOADS to Port 1 AND Port 2; then press OK"  
Scpi.Execute ("SENSe:CORRection:COLLect:ACQuire stan5")
```

```
'All standards have been measured. Save the result
```

```
Scpi.Execute ("SENS:CORR:COLL:SAVE")  
MsgBox "The calibration has been completed"
```

Perform an Unguided ECal

This VBScript program performs an Unguided Full 2-Port ECal.

The SCPI commands in this example are sent over a COM interface using the [SCPIStringParser](#) object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Unguided.vbs. [Learn how to setup and run the macro.](#)

[See Sense:Correction commands.](#)

See other SCPI Examples

```
Set pna = CreateObject("AgilentPNA835x.Application")
Set scpi = pna.ScpiStringParser
' Preset the analyzer
scpi.Execute "SYSTem:PRESet"

' Start frequency of 10 MHz
scpi.Execute "SENSe:FREQuency:STARt 10E6"

' Stop frequency of 9 GHz
scpi.Execute "SENSe:FREQuency:STOP 9E9"

' Select the preset S11 measurement
scpi.Execute "CALCulate:PARAMeter:SElect 'CH1_S11_1'"
' Read the information about the Agilent factory
' characterization data of ECal module #1 on the USB bus
module1Info = scpi.Execute("SENSe:CORREction:COLLect:CKIT:INFormation? ECAL1,CHAR0")

' Prompt for the ECal module
MsgBox "Description of ECal Module #1:" & Chr(10) & Chr(10) & module1Info & _Chr(10)
& Chr(10) & "Make port connections to the ECal module, then press enter"
' ECal full 1 port and 2 port
' Choose a Calibration Type (comment out one of these)
scpi.Execute "SENSe:CORREction:COLLect:METHod refl3"
scpi.Execute "SENSe:CORREction:COLLect:METHod SPARSOLT"
' Specify to have the PNA automatically determine which port of the
' ECal module is connected to which port of the PNA.
scpi.Execute "SENSe:CORREction:PREFerence:ECAL:ORIENTATION ON"
' Alternatively, if you are measuring at very low power levels where
' the PNA fails to sense the module's orientation, you may need to turn
' off the auto orientation and specify how the module is connected (as in
' these next two commented lines of code -- "A1,B2" would indicate Port A
' of the module is connected to Port 1 and Port B is connected to Port 2).
```

```
'scpi.Execute "SENSe:CORRection:PREFErence:ECAL:ORIEntation OFF"
'scpi.Execute "SENSe:CORRection:PREFErence:ECAL:PMAP ECAL1,'A1,B2'"
' Acquire and store the calibration terms. *OPC? causes a "+1" to be
' returned when finished. CHAR0 indicates to use the Agilent factory
' characterized data within the ECal module (as opposed to a user characterization).
x = scpi.Execute("SENSe:CORRection:COLLect:ACQuire ECAL1,CHAR0;*OPC?")
' Note: if you have set up a slow sweep speed (for example, if
' you're using a narrow IF bandwidth), and while this calibration is
' being acquired you wish to have your program perform other operations
' (like checking for the click event of a Cancel button) and you're
' NOT using the COM ScpiStringParser, you can use the optional
' ASYNchronous argument with the ACQuire command as shown here below
' instead of sending that command in the way shown above. The SCPI
' parser then will return immediately while the cal acquisition
' proceeds (i.e., the parser will NOT block-and-wait for the
' cal to finish, so you can send additional commands in the meantime).
' So you can do "*ESR?" or "*STB?" queries to monitor the status register
' bytes to see when the OPC bit gets set, which indicates the cal has
' finished. That type of OPC detection works for all of the PNA's SCPI
' parsers except the COM ScpiStringParser.
' An alternative to querying the status register, is to setup an SRQ handler
' if your IO Libraries supports that.
' When an SRQ event occurs, a call back will automatically
' "SENSe:CORRection:COLLect:ACQuire ECAL1,CHAR0,ASYNchronous;*OPC"
MsgBox "Done with calibration."
```

Perform Unknown Thru or TRL Cal

The following program performs either a 2-port SOLT Unknown Thru Cal or a 2-port TRL Cal. The 85052C Cal Kit used in this program contains both types of standards. This program can be run on 2-port or 4-port PNAs. When run on [select PNA-L models](#), a Delta Match Cal is required.

- [See Delta Match Cal example program](#)
- [See the Guided Cal commands](#)

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Unknown.vbs. [Learn how to setup and run the macro.](#)

```
Sub PerformUnknownThruOrTRLCal()  
  
Set pna = CreateObject("AgilentPNA835x.Application")  
  
Set scpi = pna.ScpiStringParser  
  
' Specify connectors for Ports 1 and 2  
  
scpi.Parse "SENS:CORR:COLL:GUID:CONN:PORT1 'APC 3.5 female'"  
scpi.Parse "SENS:CORR:COLL:GUID:CONN:PORT2 'APC 3.5 male'"  
  
'If your PNA has 3 or 4 ports, uncomment one or both of  
'these next two lines, to explicitly specify this is a 2-port cal.  
  
'scpi.Parse "SENS:CORR:COLL:GUID:CONN:PORT3 'Not used'"  
'scpi.Parse "SENS:CORR:COLL:GUID:CONN:PORT4 'Not used'"  
  
' Specify cal kit for Ports 1 and 2  
  
scpi.Parse "SENS:CORR:COLL:GUID:CKIT:PORT1 '85052C'"  
scpi.Parse "SENS:CORR:COLL:GUID:CKIT:PORT2 '85052C'"  
  
' Since the 85052C cal kit contains SOLT standards and also TRL  
' standards, these next two lines set cal and thru method.  
  
' Always send the init command before and after these two commands  
  
scpi.Parse "SENS:CORR:COLL:GUID:INIT"  
  
scpi.Parse "SENS:CORR:COLL:GUID:PATH:CMETHOD 1,2,"SOLT"  
  
scpi.Parse "SENS:CORR:COLL:GUID:PATH:TMETHOD 1,2,"UNKN"
```

```

' To set up the cal as TRL, comment the previous 'CMET' line and uncomment
' this next line. The TMETHOD is set by default
'scpi.Parse "SENS:CORR:COLL:GUID:PATH:CMETHOD 1,2,"TRL"
' Initiate the calibration
scpi.Parse "SENS:CORR:COLL:GUID:INIT"
' Query the list of ports that need delta match
retStr = scpi.Parse("SENS:CORR:COLL:GUID:DMAT:APPL:PORT?")
portList = Split(retStr, ",")
' If portList contains just one element and it's value is 0, then that indicates
' none of the ports being calibrated require delta match data.
' Note: if each testport on the PNA has it's own reference receiver (R channel),
' then delta match is never needed, so portList will always be just 0.
lowerBound = LBound(portList)
If (UBound(portList) <> lowerBound) Or (CInt( portList(lowerBound) ) <> 0) Then
' Delta match data is required for at least one port.
' For this example, we assume a Global Delta Match Cal has previously been
' performed so the Global Delta Match CalSet exists.
' The Global Delta Match CalSet is used when the APPL command is invoked
' without a specific calset ID (GUID).
scpi.Parse "SENS:CORR:COLL:GUID:DMAT:APPL"
End If
' Query the number of calibration steps
retStr = scpi.Parse("SENS:CORR:COLL:GUID:STEP?")
numSteps = CInt(retStr)
' Measure the cal standards
For i = 1 To numSteps
prompt = scpi.Parse("SENS:CORR:COLL:GUID:DESC? " & CStr(i))
retVal = MsgBox(prompt, vbOKCancel)

```

```
If retVal = vbCancel Then Exit Sub

retStr = scpi.Parse("SENS:CORR:COLL:GUID:ACQ STAN" & CStr(i) & ";*OPC?")

Next

' Compute the error coefficients and save the cal to CalSet, and turn it on

scpi.Parse "SENS:CORR:COLL:GUID:SAVE"

MsgBox "Cal is done!"

End Sub
```

Last Modified:

10-Jan-2013 Added init command before CMET and TMET
14-May-2007 MX Updated for new CMET and TMET commands

Setup Fast CW and FIFO

This example program does the following:

- Setup an A/R and B/R measurement
- Turn ON point averaging
- Set external edge triggering (commented out)
- Set FIFO and Fast CW
- Write data into FIFO data buffer
- Read FIFO data buffer

IMPORTANT - Because the IFBW is set to 600 kHz, data will NOT be sent to the FIFO after each acquisition. [Learn more.](#)

The SCPI commands in this example are sent over a COM interface using the [SCPIStringParser](#) object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as FIFO.vbs. [Learn how to setup and run the macro.](#)

[See the SCPI FIFO commands.](#)

See Other SCPI Example Programs

```
Dim returnStr
Dim app
Dim p
Dim start
Dim complete
Dim init
Dim finished
' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
```

```

Set p = app.ScpStringParser

sub Write (command)
if len(returnStr) <> 0 then
err.Raise 55,"Write","Query Unterminated"
end if

returnStr = p.parse(command)
end sub

'

sub WriteIgnoreError(command)
returnStr = p.Execute(command)
p.Parse("SYST:ERR?") ' clear error queue
end sub

'

function Read

if len(returnStr) = 0 then
err.Raise 55,"Read","Bad read"
end if

Read = returnStr

returnStr = ""

end function

' Setup and measure A/R and B/R

Write "SYST:FPRESET"

Write "DISP:WIND ON"

Write "CALC:PAR:DEF:EXT 'meas1','A/R1,0'"

Write "DISP:WIND:TRACE:FEED 'meas1'"

Write "CALC:PAR:DEF:EXT 'meas2','B/R1,0'"

Write "DISP:WIND:TRACE2:FEED 'meas2'"

' Set IFBW to 600 khz (400 thousand pts/second)

```

```

Write "SENS:BWID:RES 600khz"

' Point Averaging Count = 1

Write "SENS:AVER:MODE POINT"

Write "SENS:AVER ON"

Write "SENS:AVER:COUNT 1"

' Edge triggering - positive edge

'Write "CONT:SIGN BNCL,TIEPOSITIVE"

'Write "TRIG:SOUR EXT"

'Write "SENS:SWE:TRIG:POIN ON"

' Setup FIFO and Fast CW count

Write "SENS:SWE:MODE HOLD"

Write "SYST:FIFO ON"

Write "SYST:FIFO:DATA:CLEAR"

Write "SENS:SWE:TYPE CW"

Write "SENS:SWE:TYPE:FACW 1000000" ' set the point count to 1 million

Write "SENS:SWE:MODE SING" ' start an asynchronous acquisition.

init = now()

' Gather data

'wait until end of sweep. Timeout needs to be very large here.

Write "*OPC?" '

opcCount = Read()

Dim points

Write "SYST:FIFO:DATA:COUNT?"

points = Read()

msgbox points

' points == 2000000 ' points = 2million. Took 5 seconds to acquire

For I = 0 to 1 ' 2 iterations (2 parameters * 2 sets of 1 million)

Dim data

```

```
Write "SYST:FIFO:DATA? 1000000"  
Data = Read()  
Next  
'turn FIFO and FastCW OFF  
Write "SYST:FIFO OFF"  
Write "SENS:SWE:TYPE:FACW 0"  
  
finished = now()  
msgbox "Init =" & init & vbCrLf & "Done ="& finished
```

Last Modified:

12-Jan-2011 Replaced example with updated from TS
10-Oct-2008 MX New topic

Setup Markers using SCPI

This VBScript program does the following:

- Preset the PNA
- Return active channel number and measurement string
- Create a marker
- Set X-axis value
- Read X, Y-axis values
- Set marker to trace Min
- Read X, Y-axis values

The SCPI commands in this example are sent over a COM interface using the [SCPIStringParser](#) object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Markers.vbs. [Learn how to setup and run the macro.](#)

[See all Marker SCPI commands.](#)

See Other SCPI Example Programs

```
Dim na, vi, ret

Set na = CreateObject("AgilentPNA835x.Application")

Set vi = na.ScpIStringParser

'Get Identification String from Analyzer

ret=vi.Parse("*IDN?")

msgbox ret

'Preset PNA

ret=vi.Parse("SYST:PRES; *OPC?")

'Get Active Channel and Measurement

chan = vi.Parse("SYST:ACT:CHAN?")
```

```
meas = vi.Parse("SYST:ACT:MEAS?")
'Convert chan to a single number
chan=CStr(CInt(chan))
'Select Active Measurement
vi.Parse "CALC" + chan + ":PAR:SEL " + meas
'Turn Marker 1 on and set X value to 1 GHz
vi.Parse "CALC" + chan + ":MARK1:STAT ON"
vi.Parse "CALC" + chan + ":MARK1:X 1e9"
'Get X and Y marker values
x_val = vi.Parse("CALC" + chan + ":MARK1:X?")
y_val = vi.Parse("CALC" + chan + ":MARK1:Y?")
'Display Marker Values
msgbox "X Value = " + x_val + Chr(10) + "Y Value = " + y_val
'Use Marker 1 as a minimum search
vi.Parse "CALC" + chan + ":MARK1:FUNC:EXEC MIN"
'Get X and Y marker values
x_val = vi.Parse("CALC" + chan + ":MARK1:X?")
y_val = vi.Parse("CALC" + chan + ":MARK1:Y?")
'Display Marker Values
msgbox "X Value = " + x_val + Chr(10) + "Y Value = " + y_val
```

Last Modified:

6-Oct-2008 MX New topic

Setup Noise Figure Port Mapping

This program demonstrates how to change source and receive ports when measuring noise figure. It assumes that option 029 ("Fully Corrected Noise Figure") is installed.

If only option 028 ("Noise figure measurements using standard receivers") is installed, switching ports is simpler, since only one noise receiver selection is available.

This program can be run as a macro in the PNA. To do this, copy the code into a text editor file such as Notepad and save on the PNA hard drive as NF.vbs. [Learn how to setup and run the macro.](#)

[See the Noise figure commands.](#)

See Other SCPI Example Programs

```
option explicit
dim app, hostname, parser
set app = CreateObject("Agilentpna835x.application")
set parser = app.ScpiStringParser
' Create Noise Figure measurement
parser.Parse "*RST"
parser.Parse "CALC:PAR:DEL:ALL"
parser.Parse "CALC:CUST:DEF 'NF', 'Noise Figure Cold Source', 'NF'
"
parser.Parse "DISP:WIND:TRAC1:FEED 'NF'"
' To change from the default input/output port settings of
' source port = PNA1, receive port = PNA2, you must first
' change the noise receiver, then select the desired ports.
dim srcPort, rcvPort
' Set source=PNA port 3 and receiver=PNA port 4
srcPort = 3
```

```
rcvPort = 4
' use PNA receiver for noise measurements
parser.Parse "SENS:NOIS:REC NORMAL"
' set port mapping
parser.Parse "SENS:NOIS:PMAP " & srcPort & "," & rcvPort
' To revert back to using the noise receiver, the source
' and receive ports must be set to their default values
' BEFORE switching to the noise receiver. Otherwise, a
' SCPI "Execution error" will occur.
' restore defaults: source=PNA1, receiver=PNA2
parser.Parse "SENS:NOIS:PMAP 1,2"
' use dedicated noise receiver for noise measurements
parser.Parse "SENS:NOIS:REC NOISE"
```

Setup Phase Control

This VBScript program configures and displays Phase Sweep measurements.

The SCPI commands in this example are sent over a COM interface using the [SCPIStringParser](#) object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as RxLev.vbs. [Learn how to setup and run the macro.](#)

See Also

[Phase Control SCPI commands](#)

[About Phase Control](#)

See Other SCPI Example Programs

```
'Assume port 1 is connected to port 3

Set pna = CreateObject("AgilentPNA835x.Application")

Set SCPI = PNA.ScpIStringParser

'Create 3 trace S33, R3/C(amp),R3/C(phase)

SCPI.Parse("SYST:FPR")

SCPI.Parse("DISP:WIND:STATE ON")

SCPI.Parse("CALC:PAR:DEF 'MyMeas1',S33")

SCPI.Parse("DISP:WIND1:TRAC1:FEED 'MyMeas1'")

SCPI.Parse("CALC:PAR:SEL 'MyMeas1'")

SCPI.Parse("CALC:FORM SMIT")

SCPI.Parse("CALC:PAR:DEF 'MyMeas2',R3C,3")

SCPI.Parse("DISP:WIND1:TRAC2:FEED 'MyMeas2'")

SCPI.Parse("CALC:PAR:SEL 'MyMeas2'")

SCPI.Parse("CALC:FORM MLOG")

SCPI.Parse("CALC:PAR:DEF 'MyMeas3',R3C,3")
```

```
SCPI.Parse("DISP:WIND1:TRAC3:FEED 'MyMeas3'")
```

```
SCPI.Parse("CALC:PAR:SEL 'MyMeas3'")
```

```
SCPI.Parse("CALC:FORM PHAS")
```

```
SCPI.Parse("SENS:SWE:TYPE PHAS")
```

```
'turn on 3 and 1
```

```
SCPI.Parse("SOUR:POW1:MODE ON")
```

```
SCPI.Parse("SOUR:POW3:MODE ON")
```

```
'set port3's control parameter to R3/C
```

```
SCPI.Parse("SOUR:PHAS3:PAR 'R3/C'")
```

```
'Set port3 to PAR mode
```

```
SCPI.Parse("SOUR:PHAS3:PAR:MODE PAR")
```

```
SCPI.Parse("SOUR:PHAS3:PAR:PORT 1")
```

```
SCPI.Parse("SOUR:PHAS3:POFF:FIX 3")
```

```
SCPI.Parse("SOUR:PHAS3:STAR 0")
```

```
SCPI.Parse("SOUR:PHAS3:STOP 180")
```

Last Modified:

28-Jan-2011 MX New topic

Setup PNOP and PSAT Marker Search

This example program does the following:

- Sets up measurement for either PNOP or PSAT marker search
- Sets parameters for search
- Reads a parameter for each

See [PNOP](#) and [PSAT](#) SCPI commands.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as SearchMkr.vbs. [Learn how to setup and run the macro.](#)

See Other SCPI Example Programs

```
Dim app
Set app = CreateObject("AgilentPNA835X.Application")

Dim scpi
set scpi = app.ScpiStringParser

scpi.Execute ("SYST:FPRReset")

' View Power Out vs Power In
' Create and turn on window/channel 1

scpi.Execute ("DISPlay:WINDow1:STATE ON")

'Define a measurement name, parameter

scpi.Execute ("CALCulate1:PARAmeter:DEFine:EXT 'MyMeas',B")

'Associate ("FEED") the measurement name ('MyMeas') to WINDow (1)

scpi.Execute ("DISPlay:WINDow1:TRACe1:FEED 'MyMeas'")

scpi.Execute ("CALCulate1:PARAmeter:SELEct 'MyMeas'")

'perform power sweep
```

```

scpi.Execute ("SENSE1:SWEep:TYPE POWER")
scpi.Execute ("SOURce1:POWer:START -5")
scpi.Execute ("SOURce1:POWer:STOP 0")

'-----
'Choose marker search

resp=Msgbox ("PNOP (yes) or PSAT (no)" , 4, "PNA Marker Search Demo")

if resp=6 then

    PNOP1()

Else

    PSAT1()

End If

'-----
'PSAT marker search

Sub PSAT1()

scpi.Execute ("CALCulate1:MARKer:PSATuration:BACKoff 2")

'Read PSAT Parameter

dim answer

answer=scpi.Execute ("CALCulate1:MARKer:PSATuration:GAIN?")

wscript.echo("Gain Sat: "& answer)

End Sub

'-----
'PNOP marker search

Sub PNOP1()

scpi.Execute ("CALCulate1:MARKer:PNOP:BACKoff 2")

scpi.Execute ("CALCulate1:MARKer:PNOP:POFFset 1")

'Read PNOP Parameter

dim answer

answer=scpi.Execute ("CALCulate1:MARKer:PNOP:GAIN?")

```

```
wscript.echo("PNOP Gain: "& answer)
```

```
End Sub
```

Last Modified:

22-Feb-2010 MX New topic

Setup Receiver Leveling using SCPI

This VBScript program configures Receiver Leveling.

- Preset the PNA
- Make all receiver leveling settings

The SCPI commands in this example are sent over a COM interface using the [SCPIStringParser](#) object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as RxLev.vbs. [Learn how to setup and run the macro.](#)

[See all Receiver Leveling SCPI commands.](#)

See Other SCPI Example Programs

```
Set pna = CreateObject("AgilentPNA835x.Application")

Set SCPI = pna.ScpIStringParser

'set source port

dim srcP

srcP = "1"

'Preset PNA

SCPI.Parse "SYST:PRES"

SCPI.Parse "sour1:pow" + srcP + ":alc:mode:rec:ref 'R1'"

SCPI.Parse "sour1:pow" + srcP + ":alc:mode:rec:tol 0.02"

SCPI.Parse "sour1:pow" + srcP + ":alc:mode:rec:iter 10"

SCPI.Parse "sour1:pow" + srcP + ":alc:mode:rec:fast OFF"

SCPI.Parse "sour1:pow" + srcP + ":alc:mode:rec:ifbw 100"

SCPI.Parse "sour1:pow" + srcP + ":alc:mode:rec:offs 0"

SCPI.Parse "sour1:pow" + srcP + ":alc:mode:rec:safe:max 20"

SCPI.Parse "sour1:pow" + srcP + ":alc:mode:rec:safe:max -100"
```

```
SCPI.Parse "sour1:pow" + srcP + ":alc:mode:rec:safe ON"
```

```
'Last, enable receiver leveling
```

```
SCPI.Parse "sour1:pow" + srcP + ":alc:mode:rec ON"
```

Last Modified:

13-Feb-2009 MX New topic

Setup Sweep Parameters using SCPI

This Visual Basic program sets up sweep parameters on the Channel 1 measurement.
To run this program, you need:

- An established [GPIB interface connection](#)

See Other SCPI Example Programs

```
GPIB.Write "SYSTem:PRESet"  
'Select the measurement  
GPIB.Write "CALCulate:PARAmeter:SElect 'CH1_S11_1'"  
'Set sweep type to linear  
GPIB.Write "SENSe1:SWEep:TYPE LIN"  
  
'Set IF Bandwidth to 700 Hz  
GPIB.Write "SENSe1:BANDwidth 700"  
  
'Set Center and Span Freq's to 4 GHz  
GPIB.Write "SENSe1:FREQuency:CENTer 4ghz"  
GPIB.Write "SENSe1:FREQuency:SPAN 4ghz"  
  
'Set number of points to 801  
GPIB.Write "SENSe1:SWEep:POINTs 801"  
  
'Set sweep generation mode to Analog  
GPIB.Write "SENSe1:SWEep:GENeration ANAL"  
  
'Set sweep time to Automatic  
GPIB.Write "SENSe1:SWEep:TIME:AUTO ON"  
  
'Query the sweep time  
GPIB.Write "SENSe1:SWEep:TIME?"  
SweepTime = GPIB.Read
```

Setup the Display using SCPI

This Visual Basic program:

- Sets data formatting
- Turns ON the Trace, Title, and Frequency Annotation
- Autoscales the Trace
- Queries Per Division, Reference Level, and Reference Position
- Turn ON and set averaging
- Turn ON and set smoothing

To run this program, you need:

- An established [GPIB interface connection](#)

See Other SCPI Example Programs

```
GPIB.Write "SYSTem:PRESet"

'Select the measurement
GPIB.Write "CALCulate:PARameter:SElect 'CH1_S11_1'"

'Set the Data Format to Log Mag
GPIB.Write ":CALCulate1:FORMat MLOG"

'Turn ON the Trace, Title, and Frequency Annotation
GPIB.Write "DISplay:WINDow1:TRACe1:STATe ON"
GPIB.Write "DISplay:WINDow1:TITLe:STATe ON"
GPIB.Write "DISplay:ANNotation:FREQuency ON"

'Autoscale the Trace
GPIB.Write "DISplay:WINDow1:TRACe1:Y:Scale:AUTO"

'Query back the Per Division, Reference Level, and Reference Position
GPIB.Write "DISplay:WINDow1:TRACe1:Y:SCALE:PDIVision?"
Pdiv = GPIB.Read
GPIB.Write "DISplay:WINDow1:TRACe1:Y:SCALE:RLEVel?"
Rlev = GPIB.Read
GPIB.Write "DISplay:WINDow1:TRACe1:Y:SCALE:RPOSition?"
Ppos = GPIB.Read

'Turn ON, and average five sweeps
GPIB.Write "SENSel:AVERage:STATe ON"
GPIB.Write "SENSel:AVERage:Count 5"
```

'Turn ON, and set 20% smoothing aperture

GPIB.Write "CALCulatel:SMOothing:STATe ON"

GPIB.Write "CALCulatel:SMOothing:APERTure 20"

Show Custom Cal Windows during a Guided Calibration

This VBScript program shows how to send commands that allow you to view specific 'custom' windows, and sweep specific channels, during a UI (Cal Wizard) or remote calibration.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as CalWindow.vbs. [Learn how to setup and run the macro.](#)

These commands are used to show and sweep windows and channels:

- [SENS:CORR:COLL:DISP:WIND](#)
- [SENS:CORR:COLL:SWE:CHAN](#)
- [SENS:CORR:COLL:DISP:WIND:AOFF](#)
- [SENS:CORR:COLL:SWE:CHAN:AOFF](#)
- [SENS:CORR:COLL:GUID:PACQuire](#)

[See Other SCPI Example Programs](#)

```
Dim app

Dim scpi

' Create / Get the PNA application.

Set app = CreateObject("AgilentPNA835x.Application")

Set scpi = app.ScpiStringParser

' A comment

'Preset the analyzer

'This creates an S11 measurement in channel 1, window 1

scpi.Execute "SYST:PRreset"

' Create and turn on window 2

scpi.Execute "DISPlay:WINDow2:STATE ON"
```

'Define an S21 measurement in channel 2

```
scpi.Execute "CALCulate2:PARAMeter:DEFine:EXT 'MyMeas',S21"
```

'Associate ("FEED") the measurement name ('MyMeas') to WINDOW2

'and give the new TRACE a number (1).

```
scpi.Execute "DISPlay:WINDow2:TRACe1:FEED 'MyMeas'"
```

'The following lines are all you need in order to:

'show and sweep the custom Cal windows during a UI Calibration

'If sending ONLY these commands, make sure you know the

'correct window and channel numbers to show and sweep.

'Flag windows 1 and 2 to show during Ch1 calibration

```
scpi.Execute "SENS:CORR:COLL:DISP:WIND1 ON"
```

```
scpi.Execute "SENS:CORR:COLL:DISP:WIND2 ON"
```

'Flag channels 1 and 2 to sweep during Ch1 calibration

```
scpi.Execute "SENS1:CORR:COLL:SWE:CHAN1 ON"
```

```
scpi.Execute "SENS1:CORR:COLL:SWE:CHAN2 ON"
```

' =====

' The following code performs a remote guided Cal on Ch1.

' From a remote cal, the Cal window does not normally show and sweep

' after the previous standard has been acquired.

' This shows how to include the PACQuire (preview) to view and sweep the Cal Window.

' The Custom window also shows and sweeps due to the flag commands above.

' The flags are cleared at the end of this section.

' Specify the DUT connectors

```
scpi.Execute "sens:corr:coll:guid:conn:port1 ""APC 3.5 female"" "
```

```
scpi.Execute "sens:corr:coll:guid:conn:port2 ""APC 3.5 male"" "
' Select the Cal Kit for each port being calibrated.
scpi.Execute "sens:corr:coll:guid:ckit:port1 ""85052D"" "
scpi.Execute "sens:corr:coll:guid:ckit:port2 ""85052D"" "
' Initiate the calibration and query the number of steps
scpi.Execute "sens:corr:coll:guid:init"
numSteps = scpi.Execute("sens:corr:coll:guid:steps?")
MsgBox "Number of steps is " + CStr(numSteps)
' Measure the standards
For i = 1 to numSteps
step = "Step " + CStr(i) + " of " + CStr(numSteps)
strPrompt = scpi.Execute("sens:corr:coll:guid:desc? " + CStr(i))
'send the Preview Acquire command, then prompt
scpi.Execute "sens:corr:coll:guid:PACquire STAN" + CStr(i)
' Do NOT send any Guided Cal commands here or the cal window will not sweep
MsgBox strPrompt, vbOKOnly, step
scpi.Execute "sens:corr:coll:guid:acq STAN" + CStr(i)
Next
' Conclude the calibration
scpi.Execute "sens:corr:coll:guid:save"
MsgBox "Cal is done!"

'Remove the Custom Window flags
scpi.Execute "SENS:CORR:COLL:DISP:WIND:AOFF"
'Remove the channel sweep flags
scpi.Execute "SENS:CORR:COLL:SWE:CHAN:AOFF"
```

Last Modified:

1-Nov-2007 New topic

Perform a Sliding Load Calibration using GPIB

This Visual Basic program does a **only** the sliding load portion of a Calibration.
To run this program, you need:

- An established [GPIB interface connection](#)
- A measurement and calibration routine to call this sub-program
- STAN3 set up as a sliding load standard

[See Other SCPI Example Programs](#)

```
Sub slide()  
'Measure the sliding load for at least 5 and no more than 7 slides  
'Note that "SLSET" and "SLDONE" must be executed before the actual acquisition of a  
slide  
MsgBox "Connect Sliding Load; set to Position 1; then press OK"  
GPIB.Write "SENS:CORR:COLL SLSET"  
GPIB.Write "SENS:CORR:COLL STAN3;"  
  
MsgBox "Set Sliding Load to position 2; then press OK"  
GPIB.Write "SENS:CORR:COLL SLSET"  
GPIB.Write "SENS:CORR:COLL STAN3;"  
  
MsgBox "Set Sliding Load to position 3; then press OK"  
GPIB.Write "SENS:CORR:COLL SLDONE"  
GPIB.Write "SENS:CORR:COLL STAN3;"  
End Sub
```

Socket Client

The following C# example demonstrates how to send SCPI commands to the PNA via a TCP socket connection and how to use a TCP 'control' connection. If the command is a query, the program will read the instrument's response. You can add or replace the SCPI commands in this program with your own.

[Learn how to enable Sockets communication on the PNA.](#)

For both of the following methods, first copy the example text below into a Notepad file and name it SocketClient.cs.

To run using Microsoft Visual Studio 2003 or 2005

1. From the Visual Studio **File** menu, select **New**, then **Project**.
2. In the **New Project** window, select the following items (noting the location of the file folder it is creating for you) then click **OK**.
 - Project Type: Visual C#
 - Template: Console Application
 - Project Name: SocketClient
1. Copy SocketClient.cs into the folder that was created in the previous step.
2. In the Solution Explorer window pane, right-click **Class1.cs** (if Visual Studio 2003) or **Program.cs** (if Visual Studio 2005). Select **Delete** to delete that file.
3. In the Solution Explorer, right-click **SocketClient** , and select **Add**, then **Existing Item....**
4. Browse to select **SocketClient.cs** and click **OK**.

You should then be able to build the project, and test the resulting **SocketClient.exe** from a command prompt (shell) window.

To run using Mono

Mono is a cross-platform version of .NET. You can download a free version of Mono at <http://www.mono-project.com>. Once downloaded and installed:

1. Run the Mono command prompt (shell) window.
2. Navigate to the directory where the example SocketClient.cs is stored.

3. Type: **MCS SocketClient.cs** (builds the .exe and saves in that same folder.)
4. Type **mono SocketClient.exe** <PNA name or IP address>

This example was compiled and tested successfully with Mono version 1.1.13. It was run on a PC using the Red Hat version 9.0 distribution of the Linux operating system. It was also run on a PC using Windows XP. This program has not been tested with other versions of Mono, or on other operating systems.

To run with Agilent T&M Toolkit

Agilent T&M Toolkit 2.0 is the first version to support communication using Sockets.

Use the following to address the Sockets port: **TCPIP0::<PNA name or IP address>::5025::SOCKET**

```
using System;
using System.Net;
using System.Net.Sockets;

// This C# "Console Application" example program demonstrates SCPI
// communication with an Agilent TCP socket-enabled instrument that
// supports socket "control connections" (such as PNA network analyzers,
// which have support for control connections in their socket
// implementation as of PNA Firmware A.08.33.01).
namespace CSharpSocketClient
{
    /// <summary>
    /// The class supporting the main entry point for the application.
    /// </summary>
    class MainClass
    {
        static AsyncCallback m_pCallbackFunc;
        static string m_AsyncReply;
```

```

/// <summary>
/// The main entry point for the application.
/// </summary>

[STAThread]
static int Main(string[] args)
{
    try
    {
        if (args.Length != 1)
        {
            Console.WriteLine("");
            Console.WriteLine("Usage -- with Microsoft's .NET runtime:");
            Console.WriteLine("SocketClient servernameoraddress");
            Console.WriteLine("Example: SocketClient 192.168.0.1");
            Console.WriteLine("");
            Console.WriteLine("Usage -- with Mono's (www.mono-project.com) .NET
runtime:");
            Console.WriteLine("mono SocketClient.exe servernameoraddress");
            Console.WriteLine("Example: mono SocketClient.exe 192.168.0.1");
            return 1;
        }

        string server = args[0];

        Int32 port = 5025; // default socket port number for the PNA

        // Create the primary client socket instance

        Socket client = new Socket(AddressFamily.InterNetwork, SocketType.Stream,
ProtocolType.Tcp);

```

```
// Get the DNS IP addresses associated with the instrument.

// (if 'server' string contains the IP address rather than DNS name, this
still works)

IPHostEntry hostInfo = Dns.Resolve(server);

IPAddress[] IPAddresses = hostInfo.AddressList;

if (IPAddresses.GetLength(0) < 1)

    return 1;

// Create an endpoint to use for opening the socket connection

IPEndPoint endpoint1 = new IPEndPoint (IPAddresses[0], port);

// Open the connection to the server instrument

client.Connect(endpoint1);

if(!client.Connected)

    return 1;

// Query the instrument's ID string.

string id = Parse(client, "*IDN?");

// Clear the instrument's Status Byte

Parse(client, "*CLS");

// Enable for the OPC bit (bit 0, which has weight 1) in the instrument's
// Event Status Register, so that when that bit's value transitions from 0
to 1

// then the Event Status Register bit in the Status Byte (bit 5 of that
byte)

// will become set.

Parse(client, "*ESE 1");
```

```
// Enable for bit 5 (which has weight 32) in the Status Byte to generate an
// SRQ when that bit's value transitions from 0 to 1.

Parse(client, "*SRE 32");

// Ask the instrument for the number of a port on which a 'control'
// socket connection can be opened.

string controlPortNumStr = Parse(client, "SYSTEM:COMMunicate:TCPIP:CONTROL?
");

Int32 controlPortNum = System.Convert.ToInt32(controlPortNumStr);

// Create the client "control connection" socket instance

Socket controlClient = new Socket(AddressFamily.InterNetwork,
SocketType.Stream, ProtocolType.Tcp);

// Create an endpoint to use for opening the control connection

IPEndPoint endpoint2 = new IPEndPoint (IPAddresses[0], controlPortNum);

// Connect to the server instrument via the port number that was returned
by the instrument.

controlClient.Connect(endpoint2);

if(!controlClient.Connected)

    return 1;

// Start the control connection listening for an SRQ message.

BeginListeningForAsyncReply(controlClient);

// Now send a preset command to the instrument, accompanied by '*OPC' such
// that when that operation is complete an SRQ event will be generated
// which posts the Status Byte message on the control connection.
```

```
Parse(client, "SYSTEM:PRESet;*OPC");

// Normally at this point you would want to have your program do other
things

// right here until the SRQ callback occurs, instead of just idling here
waiting

// for it.

do {} while (m_AsyncReply == null);

// Now that the SRQ has occurred, we can issue a Device Clear via the
control connection.

Parse(controlClient, "DCL");

// The instrument will respond back with "DCL" (and linefeed character
appended

// on the end) via the control connection when it has finished processing
the

// Device Clear request. Note that this 'Response' method uses the
synchronous

// form of 'Receive', so it could potentially time out if the instrument
were

// to take a long time to process the Device Clear. So alternatively the

// 'BeginListeningForAsyncReply' could be used for this instead of
'Response'.

string deviceClearResponse = Response(controlClient);

// Close both of the socket client sessions.

controlClient.Close();

client.Close();

}

catch (ArgumentNullException e)

{
```

```
    Console.WriteLine("ArgumentNullException: {0}", e);
}
catch (SocketException e)
{
    Console.WriteLine("SocketException: {0}", e);
}

Console.WriteLine("/n Press Enter to continue...");
Console.Read();
return 0;
}

static string Parse(Socket client, string command)
{
    // Translate the passed command into ASCII and store it as a Byte array.
    Byte[] data = System.Text.Encoding.ASCII.GetBytes(command);

    // Send the command to the socket-enabled instrument.
    client.Send(data);

    // Has to be followed by a linefeed character as terminator.
    Byte[] lf = {(Byte)'\n'};
    client.Send(lf);

    Console.WriteLine("Sent: {0}", command);

    // If the message was a query (involved a question mark), receive the
    instrument response.

    if (command.IndexOf("?") >= 0)
    {
        return Response(client);
    }
}
```

```
    return "";
}

static string Response(Socket client)
{
    // Buffer to store the response bytes.
    // For simplicity of this example, we allocate just for a 256-byte maximum
    // response size.
    Byte[] data = new Byte[256];
    // Read the batch of response bytes.
    Int32 byteCount = client.Receive(data);
    // String to store the response ASCII representation.
    string responseData = System.Text.Encoding.ASCII.GetString(data, 0,
byteCount);

    Console.WriteLine("Received: {0}", responseData);

    return responseData;
}

static void BeginListeningForAsyncReply(Socket client)
{
    if (m_pCallbackFunc == null)
    {
        m_pCallbackFunc = new AsyncCallback(OnMessageReceived);
    }

    SocketPacket socPkt = new SocketPacket();

    socPkt.thisSocket = client;

    // Start asynchronously listening for a response from this client
    IAsyncResult result = client.BeginReceive (socPkt.data,
```

```

    0, socPkt.data.Length,

    SocketFlags.None,

    m_pCallbackFunc,

    socPkt);
}

class SocketPacket
{
    public Socket thisSocket;

    // For simplicity of this example, we allocate just for a 256-byte maximum
    // response size.
    public Byte[] data = new Byte[256];
}

static void OnMessageReceived(IAsyncResult asyn)
{
    SocketPacket socPkt = (SocketPacket)asyn.AsyncState ;

    Int32 byteCount = socPkt.thisSocket.EndReceive(asyn);

    m_AsyncReply = System.Text.Encoding.ASCII.GetString(socPkt.data, 0,
byteCount);

    Console.WriteLine("Received: {0}", m_AsyncReply);
}
}
}

```

Last Modified:

28-Aug-2008 Updated with new TCPIP command

Status Reporting using SCPI

This Visual Basic program demonstrates two methods of reading the analyzer's status registers:

- Polled Bit Method - reads the Limit1 register continuously.
- SRQ Method - enables an interrupt of the program when bit 6 of the status byte is set to 1. The program then queries registers to determine if the limit line failed.

To run this program, you need:

- An established [GPIB interface connection](#)
 - A form with two buttons: Poll and SRQ Method
 - A means of causing the limit line to fail, assuming it passes initially.
-

```
Private Sub Poll_Click()  
  ' POLL THE BIT METHOD  
  ' Clear status registers  
  GPIB.Write "*CLS"  
  
  'Loop FOREVER  
  Do  
    DoEvents  
    GPIB.Write "STATus:QUESTionable:LIMit1:EVENT?"  
    onn = GPIB.Read  
  Loop Until onn = 2  
  
  MsgBox "Limit 1 Failed "  
End Sub  
  
Private Sub SRQMethod_Click()  
  'SRQ METHOD  
  GPIB.Write "SYSTem:PRESet"  
  GPIB.Write "CALCulate:PARAmeter:SElect 'CH1_S11_1'"  
  'slow down the trace  
  GPIB.Write "SENS:BWID 150"  
  
  'Setup limit line  
  GPIB.Write "CALC:LIM:DATA 2,3e9,6e9,-2,-2"  
  GPIB.Write "CALC:LIMit:DISP ON"  
  GPIB.Write "CALC:LIMit:STATe ON"  
  
  ' Clear status registers.  
  GPIB.Write "*CLS;*wai"  
  ' Clear the Service Request Enable register.  
  GPIB.Write "**SRE 0"
```

```

' Clear the Standard Event Status Enable register.
GPIB.Write "*ESE 0"

' Enable questionable register, bit(10) to report to the status byte.
GPIB.Write "STATus:QUESTIONable:ENABLE 1024"

' Enable the status byte register bit3 (weight 8) to notify controller
GPIB.Write "*SRE 8"

' Enable the onGPIBNotify event
GPIB.NotifyMask = cwGPIBRQS
GPIB.Notify
End Sub

-----
Private Sub GPIB_OnGPIBNotify(ByVal mask As Integer)
' check to see what failed
' was it the analyzer?
GPIB.Write "*STB?"
onn = GPIB.Read
If onn <> 0 Then
' If yes, then was it the questionable register?
GPIB.Write "STATus:QUESTIONable:EVENT?"
onn = GPIB.Read
' Determine if the limit1 register, bit 8 is set.
If onn = 1024 Then
'if yes, then was it trace 1?
GPIB.Write "STAT:QUES:LIMIT1:EVENT?"
onn = GPIB.Read
If onn = 2 Then MsgBox ("Limit Line1 Failed")
End If
End If
End Sub

```

Transfer Data using GPIB

The following RMB examples transfer data to and from a remote PC using the [MMEM:TRANsfer](#) command.

Transferring data FROM the PNA -- TO a remote PC:

```
30      !
40      !           Set up I/O paths
50      !
60      ! Network analyzer address
70      ASSIGN @Na TO 716
75      !
77      ! File to be stored on local computer
80      ! First time -- need to create the file.
90      ! After file name, number records set to 0 (ignored by WinOS)
95      ! Use "PURGE" command to delete if desired.
100     CREATE "mytestdata.s2p",0
110     ASSIGN @File TO "mytestdata.s2p"
120     !
122     !           TRANSFER the data (download)
123     !
125     ! Analyzer has file 'testdata.s2p' in default directory
130     OUTPUT @Na;" :MMEM:TRAN? " "testdata.s2p" ""
135     !
137     ! Now read the bytes coming back from the analyzer in four steps
138     ! (1) Read and dump the first character - '#'
140     ENTER @Na USING "#,A";A$
141     !
142     ! (2) Read the next character which is the number of digits in the file size
150     ENTER @Na USING "#,A";Digit$
160     !
161     ! (3) Use the value of the number of digits to read back the file byte size
170     ! Create query string using this number of digits
180     Img$="#,"&Digit$&"A"
190     !
200     ! Byte$ holds the number of bytes in string format
210     ENTER @Na USING Img$;Byte$
220     !
225     ! (4) Read the file contents into a buffer and store the buffer contents to a
local file
230     ! Allocate a buffer for holding the data
240     ALLOCATE Dat$[VAL(Byte$)]
250     !
260     ! Set up a different image for filling the buffer
270     Img$=Byte$&"A"
280     !
290     ! Retrieve the actual file data
300     ENTER @Na USING Img$;Dat$
305     !
```

```
307 ! Now save the file locally.
310 OUTPUT @File;Dat$
320 END
```

Transferring data FROM the remote PC - TO the PNA:

```
40 ! Set up I/O paths
50 !
60 ! Network analyzer address
70 ASSIGN @Na TO 716
77 ! File to be retrieved from local computer
78 ASSIGN @File TO "mytestdata.s2p"
79 !
120 !
122 ! TRANSFER the data
123 !
230 ! Allocate a buffer for holding the data
240 ALLOCATE Dat$[26236]
250 !
260 ! Get data from the file and fill Dat$
270 ENTER @File;Dat$
280 !
325 ! Data to be transferred to analyzer file 'testupld.s2p'
325 ! in default directory.
326 !
327 ! A specific block transfer designator must follow the
328 ! file name:
329 ! '#' specifies a block transfer.
330 ! '6' specifies 6 digits to follow.
331 ! '026236' matches the buffer size allocated above
332 ! not counting <NL><END> (new line and end of file).
430 OUTPUT @Na;":MMEM:TRAN ""testupld.s2p","#6026236",Dat$
520 END
```

Last Modified:

26-Jul-2007 Added comments to example

Triggering the PNA using SCPI

To understand how to trigger the PNA using SCPI, it is very important to understand the [PNA trigger model](#). Here is a very simple explanation. These three separate functions control PNA triggering:

1. [Trigger:Source](#) - Where the trigger signals originate:

- Internal Continuous
- Internal Manual (Single)
- External - a trigger source that is connected to the PNA rear panel.

2. [Trigger:Scope](#) - what gets triggered:

- Global - each signal triggers all channels in turn.
- Channel - each signal triggers ONE channel.

3. Channel settings ([Sense<ch>:Sweep:Mode](#)) How many triggers will each channel accept before going into hold.

- HOLD - channel will not trigger.
- CONTinuous - channel triggers indefinitely.
- GROups - channel accepts the number of triggers specified with the last [SENS:SWE:GRO:COUN](#) <num>.
- **SINGLE** - channel accepts ONE trigger, then goes to HOLD.
- Point trigger [SENS1:SWE:TRIG:POINT](#)

When controlling the PNA using SCPI, a SINGLE trigger is used to ensure that a complete sweep is taken. This example demonstrates how to Single trigger the PNA using the following two methods:

- **Simplest Triggering**

- This method uses the **default** Trigger Source = Internal to send a stream of trigger signals.
- The channel is configured to ACCEPT only a single trigger signal, then HOLD ([Sense<ch>:Sweep:Mode SINGLE](#)). This is the ONLY required command.
- This method can also be used when an External trigger source sends a continuous stream of trigger signals.

- **Advanced Triggering**

- This method SENDS a single trigger from the Source, which can be from either Internal (using INIT:IMM) or External triggering.
- Each channel is configured to accept an unlimited number of triggers. This method is the only way to perform point triggering.
- When you require some channels to accept continuous triggers and other channels to accept single triggers, see [INIT:IMM Advanced](#) to learn how.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Trigger.vbs. [Learn how to setup and run the macro.](#)

Measurement setup example: This section of code can be used at the start of both methods. It sets up:

- S11 traces on two channels
- 10 data points
- Sweep time of 2 seconds - this is slow enough to allow us to watch as each trace is triggered.

```
Dim app
Dim scpi
' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
'=====
'Setup the PNA
'Preset the analyzer
scpi.Execute ("SYST:FPRreset")
' Create and turn on window/channel 1
scpi.Execute ("DISPlay:WINDow1:STATE ON")
```

```

'Define a measurement name, parameter
scpi.Execute ("CALCulate1:PARAMeter:DEFine:EXT 'MyMeas1',S11")
'Associate ("FEED") the measurement name ('MyMeas') to WINDOW (1)
scpi.Execute ("DISPlay:WINDow1:TRACe1:FEED 'MyMeas1'")
' Create and turn on window/channel 2
scpi.Execute ("DISPlay:WINDow2:STATE ON")
'Define a measurement name, parameter
scpi.Execute ("CALCulate2:PARAMeter:DEFine:EXT 'MyMeas2',S11")
'Associate ("FEED") the measurement name ('MyMeas') to WINDOW (2)
scpi.Execute ("DISPlay:WINDow2:TRACe2:FEED 'MyMeas2'")
'Set slow sweep so we can see
scpi.Execute ("SENS1:SWE:TIME 2")
scpi.Execute ("SENS2:SWE:TIME 2")
'set number of points to 10
scpi.Execute ("SENS1:SWE:POIN 10")
scpi.Execute ("SENS2:SWE:POIN 10")
'=====
' Put both channels in Hold
scpi.Execute ("SENS1:SWE:MODE HOLD")
scpi.Execute ("SENS2:SWE:MODE HOLD")
'=====
'Pick Single Send or Single Accept
resp=Msgbox ("Single Send? - Click No for Single Accept", 4, "PNA Trigger Demo")
If resp=6 Then
SingleSend()
Else
SingleAccept()
End If

```

Simple Triggering The following example sends a continuous stream of trigger signals and each PNA channel is set to ACCEPT only a signal trigger signal, then HOLD.

- This example can be used to configure External triggering where the trigger source sends a continuous stream of trigger signals. Configure the type of trigger signal that the PNA responds to using the [CONTrol:SIGNal](#) command. The command in this example sets the PNA to respond to HIGH TTL signals at the rear-panel BNC1 trigger IN connector. This command also automatically sets Trigger Source to External Trigger.
- The [TRIG SCOPE](#) (Global or Channel) setting is NOT necessary with a continuous stream of trigger signals. The example program directly controls when each channel is triggered.
- Point triggering can NOT be used with a continuous stream of trigger signals because in point triggering the channel will accept as many triggers as necessary to complete ONE full sweep. Use the [single SEND](#) example for point triggering.

```
Sub SingleAccept()  
  
'PNA sends continuous trigger signals  
  
scpi.Execute ("TRIG:SOUR IMMEDIATE")  
  
'Uncomment the following to set External triggering  
  
'scpi.Execute ("CONT:SIGN BNC1,TILHIGH")  
  
AcceptOne()  
  
End Sub  
  
  
Sub AcceptOne()  
  
'The following command makes the channel immediately sweep  
  
'*OPC? allows the measurement to complete before the controller sends another  
command  
  
scpi.Execute ("SENS1:SWE:MODE SINGLE;*OPC?")  
  
' You could do something to ch2 here before sweeping it  
  
scpi.Execute ("SENS2:SWE:MODE SINGLE;*OPC?")  
  
resp=Msgbox ("Another trigger?", 1, "PNA Trigger Demo")  
  
If resp=1 Then
```

```
AcceptOne()
```

```
End If
```

```
End Sub
```

Advanced Trigger This example section performs Single Send triggering. Here, single triggering is accomplished by SENDING one trigger signal from the Trigger source and each channel is setup to accept unlimited trigger signals. See the [INIT:IMM](#) command for more details.

- Using this method, it is possible to change [Trigger:Scope](#) to Global or Channel. Set trigger scope to channel if there is some code to execute between channel measurements. Similarly, this method can be used to set [Point triggering](#). Use this method if there is some code to execute between data point measurements.
- In addition, this method can also be used to perform External triggering if the external trigger source is capable of SENDING single triggers. See the [CONTrol:SIGNal](#) command to set the type of signal to which the PNA will respond.
- If the external source can only send a continuous stream of trigger signals, then the [Single Accept](#) section must be used.

```
Sub SingleSend()
```

```
'Set Source Internal - Manual Triggering
```

```
scpi.Execute ("TRIG:SOUR MANual")
```

```
'If using an External trigger source that is capable of
```

```
'sending SINGLE trigger signals, then uncomment the following.
```

```
'This command automatically sets trigger source to External
```

```
'scpi.Execute ("CONT:SIGN BNCl,TILHIGH")
```

```
'Setup Trigger Scope
```

```
'WHAT gets triggered
```

```
'Pick one using comments
```

```
'Set Channel triggering
```

```
'scpi.Execute ("TRIG:SCOPE CURRent")
```

```
'Set Global triggering (Default)
```

```
scpi.Execute ("TRIG:SCOPE ALL")
```

```
'Set Channel Settings

'The channels respond to UNLIMITED trigger signals (Default)

scpi.Execute ("SENS1:SWE:MODE CONTinuous")

scpi.Execute ("SENS2:SWE:MODE CONTinuous")

'To do Point trigger on one or more channels, uncomment the following.

'Point trigger automatically sets Trig:Scope to Current/Channel

'scpi.Execute ("SENS1:SWE:TRIG:POINT ON")

'scpi.Execute ("SENS2:SWE:TRIG:POINT ON")

IntTrig()

End Sub

Sub IntTrig()

'If External triggering, replace this Sub with code

'to single trigger the External Trig Source

Dim resp

'*OPC? allows the measurement to complete before the controller sends another
command

scpi.Execute ("INITiate:IMMediate;*OPC?")

resp=Msgbox ("Another trigger?", 1, "PNA Trigger Demo")

If resp=1 Then

IntTrig()

End If

End Sub
```

Last modified:

8-Mar-2013	Slight mods
18-Jun-2007	Updated with Sens:Swe:Mode Single
June 6, 2007	Changed order and wording
April 24, 2007	Updated with links
Oct. 5, 2006	New topic

Perform an Unguided Cal on Multiple Channels

This VBScript program performs an Unguided Calibration simultaneously on two channels.

This could be used in the following cases:

- If you need more than the current number of data points per trace, so the additional points must be added to a different channel.
- If you need several channels with independent settings, but you want to calibrate all channels with a minimal number of standard connections. This would be especially critical for on wafer calibration.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Unguided.vbs. [Learn how to setup and run the macro.](#)

```
Dim app

Dim scpi

Dim NumberOfActiveChannels

NumberOfActiveChannels = 2

' Create / Get the PNA application.

Set app = CreateObject("AgilentPNA835x.Application")

Set scpi = app.ScpiStringParser

' Query the list of connectors that the PNA system recognizes

scpi.Execute("SYST:PRES")

'Wait for successful preset before continuing

done=scpi.Execute("*OPC?")

'The following section sets up 2 channels with different frequency ranges

scpi.Execute("DISP:WIND1:STATE OFF")

'Reset Windows

scpi.Execute("DISP:WIND1:STATE ON")

scpi.Execute("DISP:WIND2:STATE ON")
```

```
'  
  
' Assign a measurement to the first window  
scpi.Execute("CALC1:PAR:DEF:EXT 'Meas1', S21")  
scpi.Execute("DISP:WIND1:TRAC1:FEED 'Meas1'")  
  
'Assign a measurement to the second window  
scpi.Execute("CALC2:PAR:DEF:EXT 'Meas2', S21")  
scpi.Execute("DISP:WIND2:TRAC1:FEED 'Meas2'")  
  
'Set up two channels with independent parameters  
scpi.Execute("SENS1:FREQ:SPAN 1e9")  
scpi.Execute("SENS2:FREQ:SPAN 1e6")  
  
'Wait for changes before continuing  
done=scpi.Execute("*OPC?")  
  
'  
  
'This section sets the calibration kits for channel 1 and channel 2  
'Select a trace from channel 1 and set calibration type and cal kit  
scpi.Execute("CALC1:PAR:SEL 'Meas1'")  
scpi.Execute("SENS1:CORR:COLL:METH SPARSOLT")  
scpi.Execute("SENS1:CORR:COLL:CKIT 2") '85056D for default settings  
  
'Same standards for forward and reverse direction  
scpi.Execute("SENS1:CORR:TST OFF")  
  
'Select a trace from channel 2 and set calibration type and cal kit  
scpi.Execute("CALC2:PAR:SEL 'Meas2'")  
scpi.Execute("SENS2:CORR:COLL:METH SPARSOLT")  
scpi.Execute("SENS2:CORR:COLL:CKIT 2") '85056D for default settings  
  
'Same standards for forward and reverse direction  
scpi.Execute("SENS2:CORR:TST OFF")  
  
  
'Set both channels to manual triggering
```

```
scpi.Execute("INIT1:CONT OFF")
scpi.Execute("INIT2:CONT OFF")
'
'The following assumes female port connector on port 1
' and male port connector on port 1
'Step through all active channels and calibrate and measure all standards.
scpi.Execute("SENS1:CORR:SFOR ON") 'Set acquisition to forward
scpi.Execute("SENS2:CORR:SFOR ON") 'Set acquisition to forward
MsgBox("Connect OPEN standard to port 1")
For CurrentChannel = 1 To NumberOfActiveChannels
scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & "'")
scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL stan1")
done= scpi.Execute("*OPC?")
Next

MsgBox("Connect SHORT standard to port 1")
For CurrentChannel = 1 To NumberOfActiveChannels
scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & "'")
scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL stan2")
done=scpi.Execute("*OPC?")
Next

MsgBox("Connect LOAD standard to port 1")
For CurrentChannel = 1 To NumberOfActiveChannels
scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & "'")
scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL stan3")
done=scpi.Execute("*OPC?")
Next
```

```
scpi.Execute("SENS1:CORR:SFOR OFF") 'Set acquisition to reverse
scpi.Execute("SENS2:CORR:SFOR OFF") 'Set acquisition to forward

MsgBox("Connect OPEN standard to port 2")
For CurrentChannel = 1 To NumberOfActiveChannels
scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & "'")
scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL stan1")
done=scpi.Execute("*OPC?")
Next

MsgBox("Connect SHORT standard to port 2")
For CurrentChannel = 1 To NumberOfActiveChannels
scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & "'")
scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL stan2")
done=scpi.Execute("*OPC?")
Next

MsgBox("Connect LOAD standard to port 2")
For CurrentChannel = 1 To NumberOfActiveChannels
scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & "'")
scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL stan3")
done=scpi.Execute("*OPC?")
Next
'
'Measure thru standard for all channels in both forward and reverse direction
MsgBox("Connect THRU between ports 1 and 2")
scpi.Execute("SENS1:CORR:SFOR ON") 'Set acquisition to forward
scpi.Execute("SENS2:CORR:SFOR ON") 'Set acquisition to forward
```

```
For CurrentChannel = 1 To NumberOfActiveChannels
scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & "'")
scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL stan4")
done=scpi.Execute("*OPC?")
Next

scpi.Execute("SENS1:CORR:SFOR OFF") 'Set acquisition to reverse
scpi.Execute("SENS2:CORR:SFOR OFF") 'Set acquisition to reverse
For CurrentChannel = 1 To NumberOfActiveChannels
scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & "'")
scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL stan4")
done=scpi.Execute("*OPC?")
Next

For CurrentChannel = 1 To NumberOfActiveChannels
scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & "'")
scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL:SAVE")
done=scpi.Execute("*OPC?")
Next

'Set both channels to continuous triggering
scpi.Execute("INIT1:CONT ON")
scpi.Execute("INIT2:CONT ON")
```

Upload and Download a Segment List

This VBScript program creates two segments, then uploads the segment data to the PNA.

The second part [downloads the segment list from the PNA](#).

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Unguided.vbs. [Learn how to setup and run the macro](#).

[See all Segment SCPI commands](#).

Create and Upload a Segment List

```
Option Explicit

Dim app

Set app = CreateObject("AgilentPNA835x.Application")

' Preset the PNA

app.Preset

Dim scpi

Set scpi = app.ScpiStringParser

' In case of a measurement receiver PNA like N5264A
' which has no source ports, "SOURCE:CAtalog?" will
' return an empty list (just a pair of quotation marks)

Dim srcPortNames

srcPortNames = Split( scpi.Execute("SOURCE:CAtalog?"), ",")

Dim numberOfSrcPorts

If Left( srcPortNames(0), 2 ) = Chr(34) & Chr(34) Then

    numberOfSrcPorts = 0

Else

    numberOfSrcPorts = UBound(srcPortNames) + 1
```

```

End If

' Building up a string consisting of the sweep segment data
' we want to set up. This example will create two segments.
Dim segData

' Set state of first segment to be ON (1 = ON, 0 = OFF),
' 101 points, start freq of 10 MHz, stop freq of 1 GHz
segData = "1,101,10E6,1E9"

' If you want to include one or more of: IFbandwidth, Dwell Time
' or Port Power, remove the comments from these next two lines
'TurnOnOptions 1 'Call the subroutine
'segData = AddOptionalSettings(segData, numberOfSrcPorts)
' Set state of second segment to be ON, 201 points,
' start freq of 1 GHz, stop freq of 3 GHz
segData = segData & ",1,201,1E9,3E9"

' Uncomment this line below only if you uncommented the
' AddOptionalSettings line above for the first segment.
'segData = AddOptionalSettings(segData, numberOfSrcPorts)

Const numSegs = 2

' Upload our segment list to the channel
scpi.Execute "SENSE1:SEGMENT:LIST SSTOP," & numSegs & "," & segData

' Set segment sweep type on Channel 1
scpi.Execute "SENSE1:SWEep:TYPE SEGMENT"

' Having the PNA display the segment sweep table for the channel
scpi.Execute "DISPlay:WINDow1:TABLE SEGMENT"

Sub TurnOnOptions(ByVal chan)

    scpi.Execute "SENSE"&chan&":SEGMENT:BWIDth:CONTROL ON"

    scpi.Execute "SENSE"&chan&":SEGMENT:SWEep:TIME:CONTROL ON"

    scpi.Execute "SENSE"&chan&":SEGMENT:POWER:CONTROL ON"

```

```

' Turning off coupling allows power to vary per each port
scpi.Execute "SOURCE"&chan&":POWER:COUPLE OFF"

End Sub

Function AddOptionalSettings(ByVal inStr, ByVal numSrcPorts)

' Specifying 1 kHz IF bandwidth and Dwell Time of 0
inStr = inStr & ",1E3,0"

' -10 dBm power for each of the source ports

Dim i

For i = 0 To numSrcPorts - 1

    inStr = inStr & ",-10"

Next

AddOptionalSettings = inStr

End Function

```

[Download a Segment List](#)

This example assumes that the active trace is in Window 1

```

Option Explicit

Dim app

Set app = CreateObject("AgilentPNA835x.Application")

Dim scpi

Set scpi = app.ScpiStringParser

' Set the display-active channel's sweep type to segment sweep
' (if the PNA's currently active measurement window doesn't
' contain any traces, this querying for active channel will
' result in a SCPI error which scpi.Parse will trap and throw)

Dim chan

chan = CLng( scpi.Parse("SYSTEM:ACTIVE:CHANNEL?") )

scpi.Execute "SENSE"&chan&":SWEep:TYPE SEGment"

```

```

' Having the PNA display the segment sweep table for the channel
scpi.Execute "DISPlay:WINDow1:TABLE SEGMENT"

' Get the total number of segments
Dim numSegs

numSegs = CLng( scpi.Execute("SENSe"&chan&":SEGMENT:COUNT?") )

' Read the segment listing
Dim segDataStr

segDataStr = scpi.Execute("SENSe"&chan&":SEGMENT:LIST?")

Dim segData

segData = Split(segDataStr, ",")

' Get upper bound of the array of data values
' (lower bound of array resulting from VB 'Split' function is 0)
Dim segArrayUB

segArrayUB = UBound(segData)

Dim numDataElementsPerSeg

numDataElementsPerSeg = (segArrayUB + 1) / numSegs

WScript.Echo "Number of segments = " & numSegs

WScript.Echo "Number of data values per segment = " & numDataElementsPerSeg

Dim segInfStr

segInfStr = "Segment 1: state = " & CBool(segData(0))

segInfStr = segInfStr & ", num points = " & CLng(segData(1))

segInfStr = segInfStr & ", start freq = " & CDbL(segData(2))

segInfStr = segInfStr & ", stop freq = " & CDbL(segData(3))

segInfStr = segInfStr & ", IFBW = " & CDbL(segData(4))

segInfStr = segInfStr & ", dwell time = " & CDbL(segData(5))

' In case of a measurement receiver PNA like N5264A
' which has no source ports, "SOURce:CATalog?" will
' return an empty list

```

```

Dim srcPortNames

srcPortNames = Split( scpi.Execute("SOURCE"&chan&":CATalog?"), ",")

Dim srcPortNamesUB

srcPortNamesUB = UBound(srcPortNames)

' First source port name will be preceded by a quotation mark
' and the last name will be followed by one of those, so stripping
' those off now.

srcPortNames(0) = Right( srcPortNames(0), Len(srcPortNames(0)) - 1 )

srcPortNames(srcPortNamesUB) = Left( srcPortNames(srcPortNamesUB),
InStrRev(srcPortNames(srcPortNamesUB), Chr(34)) - 1 )

Dim firstPortIndex

firstPortIndex = 6

Dim lastPortIndex

lastPortIndex = numDataElementsPerSeg - 1

Dim j

For j = firstPortIndex To lastPortIndex

    segInfStr = segInfStr & ", " & srcPortNames(j - firstPortIndex) & " power = "
& Cdbl(segData(j))

Next

WScript.Echo segInfStr

```

Last Modified:

29-Apr-2009 MX New topic

Uploading a Source Power Cal using SCPI

Programming the PNA using COM or using SICL/VISA over LAN (as in this example) leaves the PNA free to control GPIB devices as needed. This Visual Basic program demonstrates:

- Uploading a source power calibration of Port 2 for Channel 1.
- Reading the calibration data.

Learn more about [Power Calibrations](#)

Other SCPI Example Programs

To run this program, you need:

- Your PC and PNA both connected to a LAN (if using VISA LAN server / client).
- The SICL and VISA components of Agilent's I/O Libraries software installed on your PC (both are included when you install the software, unless you already have another vendor's VISA installed. Then specify Full SICL and VISA installation to overwrite the other vendor's VISA).
- The module visa32.bas added to your VB project.
- A form with two buttons: cmdRun and cmdQuit.
- A VISA interface configured on your remote PC to control the PNA. This could be GPIB interface or a [VISA LAN Client](#).

```
'Session to VISA Default Resource Manager
Private defRM As Long
'Session to PNA
Private vipNA As Long
'VISA function status return code
Private status As Long
Private Sub Form_Load()
defRM = 0
End Sub
Private Sub cmdRun_Click()

' String to receive data from the PNA.
' Dimensioned large enough to receive scalar comma-delimited values
' for 21 frequency points (20 ASCII characters per point)
Dim strReply As String * 420
Dim strPower As String, strCalPower As String
Dim strStimulus, strCalValue
Dim strResult As String
```

```

' Open the VISA default resource manager
status = viOpenDefaultRM(defRM)
If (status < VI_SUCCESS) Then HandleVISAError

' Open a session (vipNA) to the PNA at "address 16" on the VISA
' interface configured as "GPIB0" on this PC.
status = viOpen(defRM, "GPIB0::16::INSTR", 0, 0, vipNA)
If (status < VI_SUCCESS) Then HandleVISAError

' Set the number of sweep points to 2 on Channel 1.
status = myGPIBWrite(vipNA, "SENS1:SWE:POIN 2")
If (status < VI_SUCCESS) Then HandleVISAError

' Ensure there's currently no source power cal on for this channel and port.
status = myGPIBWrite(vipNA, "SOUR1:POW2:CORR OFF")
If (status < VI_SUCCESS) Then HandleVISAError

' Specify if the cal power level is offset (positive value for a gain, negative
' value for a loss) from the PNA port power setting on the channel when no source
' power cal is active. This is to account for components between the PNA test
' port and cal reference plane. In this example, let's set up our calibration
' at the output of an amplifier with 15 dB gain.
status = myGPIBWrite(vipNA, "SOUR1:POW2:CORR:OFFS 15 DB")
If (status < VI_SUCCESS) Then HandleVISAError

' Prepare for doing data transfer in ASCII format.
status = myGPIBWrite(vipNA, "FORM:DATA ASCII")
If (status < VI_SUCCESS) Then HandleVISAError

' Send our source power correction data to the PNA. For purpose of simplicity
' in this example, we'll set up for no correction (0) at our start stimulus and
' 0.5 dB at our stop stimulus (recall that our sweep currently has just 2 points).
status = myGPIBWrite(vipNA, "SOUR1:POW2:CORR:DATA 0,0.5")
If (status < VI_SUCCESS) Then HandleVISAError

' Set the number of sweep points to 21 on Channel 1.
status = myGPIBWrite(vipNA, "SENS1:SWE:POIN 21")
If (status < VI_SUCCESS) Then HandleVISAError

' Read the fixed power level for this port on Channel 1.
status = myGPIBWrite(vipNA, "SOUR1:POW2:LEV?")
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBRead(vipNA, strReply)
If (status < VI_SUCCESS) Then HandleVISAError
strPower = strReply

' Turn the source power cal on.
status = myGPIBWrite(vipNA, "SOUR1:POW2:CORR ON")
If (status < VI_SUCCESS) Then HandleVISAError

' Again read the fixed power level for this port on Channel 1

```

```

' (with our calibration turned on, this should now include the 15 dB offset
' we indicated our power amplifier provides).
status = myGPIBWrite(viPNA, "SOUR1:POW2:LEV?")
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBRead(viPNA, strReply)
If (status < VI_SUCCESS) Then HandleVISAError
strCalPower = strReply

' Read the stimulus values from Channel 1.
status = myGPIBWrite(viPNA, "SENS1:X?")
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBRead(viPNA, strReply)
If (status < VI_SUCCESS) Then HandleVISAError

' Tokenize the reply string into an array containing the values
strStimulus = Split(strReply, ",")

' Read back the source power correction data, now interpolated for 21 points
status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:DATA?")
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBRead(viPNA, strReply)
If (status < VI_SUCCESS) Then HandleVISAError

' Tokenize the reply string into an array containing the values
strCalValue = Split(strReply, ",")

' Print the data using a message box (here, Chr returns the ASCII characters
' for Tab (9) and Linefeed (10)).
strResult = "PNA port power = " & Val(strPower) & Chr(10)
strResult = strResult & "Power at reference plane = " & Val(strCalPower) & Chr(10)
Chr(10)
strResult = strResult & "Stimulus" & Chr(9) & Chr(9) & "Cal Value" & Chr(10)
For i = 0 To UBound(strStimulus)
    strResult = strResult & Val(strStimulus(i)) & Chr(9) & Val(strCalValue(i)) &
Chr(10)
Next
MsgBox strResult
End Sub
Private Sub cmdQuit_Click()

' Close the resource manager session (which also closes
' the session to the PNA).
If defRM <> 0 Then Call viClose(defRM)

' End the program
End
End Sub
Private Function myGPIBWrite(ByVal viHandle As Long, ByVal strOut As String) As Long

' The "+ Chr$(10)" appends an ASCII linefeed character to the
' output, for terminating the write transaction.
myGPIBWrite = viPrintf(viHandle, strOut + Chr$(10), 0)

```

```
End Function
Private Function myGPIBRead(ByVal viHandle As Long, strIn As String) As Long
myGPIBRead = viVScanf(viHandle, "%t", strIn)

' Remove trailing linefeed character
If Right(strIn, 1) = Chr(10) Then strIn = Left(strIn, Len(strIn) - 1)
End Function
Sub HandleVISAError()
Dim strVisaErr As String * 200
Call viStatusDesc(defRM, status, strVisaErr)
MsgBox "*** Error : " + strVisaErr, vbExclamation

' Close the resource manager session (which also closes
' the session to the PNA).
If defRM <> 0 Then Call viClose(defRM)
End
End Sub
```

GPIB Fundamentals

The General Purpose Interface Bus (GPIB) is a system of hardware and software that allows you to control test equipment to make measurements quickly and accurately. This topic contains the following information:

- [The GPIB Hardware Components](#)
- [The GPIB / SCPI Programming Elements](#)
- [Specifications](#)
- [GPIB Interface Capability Codes](#)

Note: All of the topics related to programming assume that you already know how to program, preferably using a language that can control instruments.

Other Topics about GPIB Concepts

The GPIB Hardware Components

The system bus and its associated interface operations are defined by the IEEE 488 standard. The following sections list and describe the main pieces of hardware in a GPIB system:

Early PNA models had only ONE GPIB connector. These models could control other GPIB devices using one of, or a combination of, the following methods:

- Use the SCPI [SYST:COMM:GPIB:RDEV:](#) commands.
- Use VISA or SICL over LAN to accomplish this. See an [example](#).
- Use [USB / GPIB Interface](#)

Note: Current PNA models have dedicated Controller and Talker/Listener GPIB ports. [See how to configure these ports.](#)

Controllers

Controllers specify the instruments that will be the talker and listener in a data exchange. The controller of the bus must have a GPIB interface card to communicate on the GPIB.

- The **Active Controller** is the computer or instrument that is currently controlling data exchanges.

- The **System Controller** is the only computer or instrument that can take control and give up control of the GPIB to another computer or instrument, which is then called the active controller.

Talker / Listener Instruments and GPIB Addresses

- **Talkers** are instruments that can be addressed to send data to the controller.
- **Listeners** are instruments that can be addressed to receive a command, and then respond to the command. All devices on the bus are required to listen.

Every GPIB instrument must have its own unique address on the bus. The PNA address (default = 716) consists of two parts:

1. **The Interface select code** (typically 7) indicates which GPIB port in the system controller is used to communicate with the device.
2. **The primary address** (16) is set at the factory. You can change the primary address of any device on the bus to any number between 0 and 30. To change the analyzer address click [System / Configure / SICL-GPIB](#).

A **secondary address** is sometimes used to allow access to individual modules in a modular instrument system, such as a VXI mainframe. The PNA does NOT have a secondary address.

Cables

GPIB Cables are the physical link connecting all of the devices on the bus. There are eight data lines in a GPIB cable that send data from one device to another. There are also eight control lines that manage traffic on the data lines and control other interface operations.

You can connect instruments to the controller in any arrangement with the following limitations:

- Do not connect more than 15 devices on any GPIB system. This number can be extended with the use of a bus extension.
- Do not exceed a total of 20 meters of total cable length or 2 meters per device, whichever is less.
- Avoid stacking more than three connectors on the back panel of an instrument. This can cause unnecessary strain on the rear-panel connector.

The GPIB / SCPI Programming Elements

The following software programming elements combine to become a GPIB program:

- [GPIB / SCPI Commands](#)
- [Programming Statements](#)

- [Instrument Drivers](#)

GPIB Commands

The GPIB command is the basic unit of communication in a GPIB system. The analyzer responds to three types of GPIB commands:

1. IEEE 488.1 Bus-management Commands

These commands are used primarily to tell some or all of the devices on the bus to perform certain interface operations.

All of the functions that can be accomplished with these commands can also be done with IEEE 488.2 or SCPI commands. Therefore, these commands are not documented in this Help system. For a complete list of IEEE 488.1 commands refer to the IEEE 488 standard. **Examples** of IEEE 488.1 Commands

- `CLEAR` - Clears the bus of any pending operations
- `LOCAL` - Returns instruments to local operation

2. IEEE 488.2 Common Commands

These commands are sent to instruments to perform interface operations. An IEEE 488.2 common command consists of a single mnemonic and is preceded by an asterisk (*). Some of the commands have a query form which adds a "?" after the command. These commands ask the instrument for the current setting. See a complete list of the [Common Commands](#) that are recognized by the analyzer.

Examples of IEEE 488.2 Common Commands

- `*OPC` - Operation Complete
- `*RST` - Reset
- `*OPT?` - Queries the option configuration

3. SCPI Commands

The Standard Commands for Programmable Instruments (SCPI) is a set of commands developed in 1990. The standardization provided in SCPI commands helps ensure that programs written for a particular SCPI instrument are easily adapted to work with a similar SCPI instrument. SCPI commands tell instruments to do device specific functions. For example, SCPI commands could tell an instrument to make a measurement and output data to a controller. **Examples** of SCPI Commands:

```
CALCULATE:AVERAGE:STATE ON
```

```
SENSE:FREQUENCY:START?
```

For more information on SCPI:

- [The Rules and Syntax of SCPI Commands](#) provides more detail of the SCPI command structure.
- [SCPI Command Tree](#) is a complete list of the SCPI commands for the analyzer

Programming Statements

SCPI commands are included with the language specific I/O statements to form program statements. The programming language determines the syntax of the programming statements. SCPI programs can be written in a variety of programming languages such as VEE, HP BASIC, or C++. **Example** of a Visual Basic statement:

- `GPIB.Write "SOURCE:FREQUENCY:FIXED 1000 MHZ"`

[Note about examples](#)

Instrument Drivers

Instrument drivers are subroutines that provide routine functionality and can be reused from program to program. GPIB industry leaders have written standards for use by programmers who develop drivers. When programmers write drivers that comply with the standards, the drivers can be used with predictable results. To comply with the standard, each instrument driver must include documentation describing its functionality and how it should be implemented.

GPIB Specifications

Interconnected devices - Up to 15 devices (maximum) on one contiguous bus.

Interconnection path - Star or linear (or mixed) bus network, up to 20 meters total transmission path length or 2 meters per device, whichever is less.

Message transfer scheme - Byte-serial, bit-parallel, asynchronous data transfer using an interlocking 3-wire handshake.

Maximum data rate - 1 megabyte per second over limited distances, 250 to 500 kilobytes per second typical maximum over a full transmission path. The devices on the bus determine the actual data rate.

Address capability - Primary addresses, 31 Talk and 31 Listen; secondary addresses, 961 Talk and 961 Listen. There can be a maximum of 1 Talker and up to 14 Listeners at a time on a single bus. See also previous section on [GPIB addresses](#).

GPIB Interface Capability Codes

The IEEE 488.1 standard requires that all GPIB compatible instruments display their interface capabilities on the rear panel using codes. The codes on the analyzer, and their related descriptions, are

listed below:

SH1 full source handshake capability

AH1 full acceptor handshake capability

T6 basic talker, serial poll, no talk only, unaddress if MLA (My Listen Address)

TEO no extended talker capability

L4 basic listener, no listen only, unaddress if MTA (My Talk Address)

LEO no extended listener capability

SR1 full service request capability

RL1 full remote / local capability

PPO **no parallel poll capability**

DC1 full device clear capability

DT1 full device trigger capability

C1 system controller capability

C2 send IFC (Interface Clear) and take charge controller capability

C3 send REN (Remote Enable) controller capability

C4 respond to SRQ (Service Request)

Last Modified:

30-Oct-2008 Removed legacy content

The Rules and Syntax of SCPI

Most of the commands used for controlling instruments on the GPIB are SCPI commands. The following sections will help you learn to use SCPI commands in your programs.

[Branches on the Command Tree](#)

[Command and Query](#)

[Multiple Commands](#)

[Command Abbreviation](#)

[Bracketed \(Optional\) Keywords](#)

[Vertical Bars \(Pipes\)](#)

[MIN and MAX Parameters](#)

Other Topics about GPIB Concepts

Branches on the Command Tree

All major functions on the analyzer are assigned keywords which are called ROOT commands. (See GPIB Command Finder for a list of SCPI root commands). Under these root commands are branches that contain one or more keywords. The branching continues until each analyzer function is assigned to a branch. A root command and the branches below it is sometimes known as a subsystem.

For example, under [SOURCE:POWER](#) are several branch commands.

Sometimes the same keyword, such as STATE, is used in several branches of the command tree. To keep track of the current branch, the analyzer's command parser uses the following rules:

- **Power On and Reset** - After power is cycled or after *RST, the current path is set to the root level commands.
- **Message Terminators** - A message terminator, such as a <NL> character, sets the current path to the root command level. Many programming language output statements send message terminators automatically. Message terminators are described in Sending Messages to the Analyzer.
- **Colon (:)** - When a colon is between two command keywords, it moves the current path down one level in the command tree. For example, the colon in :SOURCE:POWER specifies that POWER is one level below SOURCE. When the colon is the first character of a command, it specifies that the following keyword is a root level command. For example, the colon in :SOURCE specifies that source is a root level command.

Note: You can omit the leading colon if the command is the first of a new program line. For example, the following two commands are equivalent:

```
SOUR:POW:ATT:AUTO
:SOUR:POW:ATT:AUTO
```

- **<WSP>** - Whitespace characters, such as <tab> and <space>, are generally ignored. There are two important exceptions:
 - Whitespace inside a keyword, such as :CALC ULATE, is not allowed.
 - Most commands end with a parameter. You must use whitespace to separate these ending parameters from commands. **Always refer to the command documentation**. In the following example, there is whitespace between STATE and ON.

```
CALCULATE1:SMOOTHING:STATE ON
```

- **Comma (,)** - If a command requires more than one parameter, you must separate adjacent parameters using a comma. For example, the SYSTEM:TIME command requires three values to set the analyzer clock: one for hours, one for minutes, and one for seconds. A message to set the clock to 8:45 AM would be SYSTEM:TIME 8,45,0. Commas do not affect the current path.
- **Semicolon(;)** - A semicolon separates two commands in the same message without changing the current path. See [Multiple Commands](#) later in this topic.
- **IEEE 488.2 Common Commands** - Common commands, such as *RST, are not part of any subsystem. An instrument interprets them in the same way, regardless of the current path setting.

Command and Query

A SCPI command can be an Event command, Query command (a command that asks the analyzer for information), or both. The following are descriptions and examples of each form of command. GPIB Command Finder lists every SCPI command that is recognized by the analyzer, and its form.

Form

Event commands - cause an action to occur inside the analyzer.

Query commands - query only; there is no associated analyzer state to set.

Command and query - set or query an analyzer setting. The query form appends a question mark (?) to the set form

Examples

```
:INITIATE:IMMEDIATE
```

```
:SYSTEM:ERROR?
```

```
:FORMAT:DATA ! Command
```

```
:FORMAT:DATA? ! Query
```

Multiple Commands

You can send multiple commands within a single program message. By separating the commands with semicolons the current path does not change. The following examples show three methods to send two commands:

1. **Two program messages:**

```
SOURCE:POWER:START 0DBM
SOURCE:POWER:STOP 10DBM
```

2. **One long message.** A colon follows the semicolon that separates the two commands causing the command parser to reset to the root of the command tree. As a result, the next command is only valid if it includes the entire keyword path from the root of the tree:

```
SOURCE:POWER:START 0DBM;:SOURCE:POWER:STOP 10DBM
```

3. **One short message.** The command parser keeps track of the position in the command tree. Therefore, you can simplify your program messages by including only the keyword at the same level in the command tree.

```
SOURCE:POWER:START 0DBM;STOP 10DBM
```

Common Commands and SCPI Commands

You can send Common commands and SCPI commands together in the same message. (For more information on these types of commands see [GP-IB Fundamentals](#).) As in sending multiple SCPI commands, you must separate them with a semicolon.

Example of Common command and SCPI commands together

```
*RST;SENSE:FREQUENCY:CENTER 5MHZ;SPAN 100KHZ
```

Command Abbreviation

Each command has a long form and an abbreviated short form. The syntax used in this Help system use uppercase characters to identify the short form of a particular keyword. The remainder of the keyword is lower case to complete the long form.

```
SOUR - Short form
SOURCE - Long form
```

Either the complete short form or complete long form must be used for each keyword. However, the keywords used to make a complete SCPI command can be a combination of short form and long form.

The following is **unacceptable** - The first three keywords use neither short or long form.

```
SOURc:Powe:Atten:Auto on
```

The following is **acceptable** - All keywords are either short form or long form.

```
SOUR:POWer:ATT:AUTO on
```

In addition, the analyzer accepts lowercase and uppercase characters as equivalent as shown in the following equivalent commands:

```
source:POW:att:auto ON  
Source:Pow:Att:Auto on
```

Optional [Bracketed] Keywords

You can omit some keywords without changing the effect of the command. These optional, or default, keywords are used in many subsystems and are identified by brackets in syntax diagrams.

Example of Optional Keywords

The HCOpy subsystem contains the optional keyword IMMEDIATE at its first branching point. Both of the following commands are equivalent:

```
"HCOpy:IMMEDIATE"  
"HCOpy"  
The syntax in this Help system looks like this:  
HCOpy[:IMMEDIATE]
```

Vertical Bars | Pipes

Vertical bars, or "pipes", can be read as "**or**". They are used in syntax diagrams to separate alternative parameter options.

Example of Vertical Bars:

```
SOURce:POWer:ATTenuation:AUTO <on|off>
```

Either ON or OFF is a valid parameter option.

MIN and MAX Parameters

The special form parameters "MINimum" and "MAXimum" can be used with commands that specify single frequency (Hz) and time (seconds) as noted in the command documentation. **Note:** Also with these commands, kHz, MHz, and GHz are accepted as suffixes/units.

The short form (min) and long form (minimum) of these two keywords are equivalent.

- **MAX**imum refers to the largest value that the function can currently be set to
- **MIN**imum refers to the smallest value that the function can currently be set to.

For example, the following command sets the start frequency to the smallest value that is currently possible:

```
SENS:FREQ:START MIN
```

In addition, the max and min values can also be queried for these commands.

For example, the following command returns the smallest value that Start Frequency can currently be set to:

```
SENS:FREQ:START? MIN
```

An error will be returned if a numeric parameter is sent that exceeds the MAX and MIN values.

For example, the following command will return an "Out of range" error message.

```
SENS:FREQ:START 1khz
```

Last Modified:

10-Jul-2007 Removed image

Configure for GPIB, SCPI, and SICL

The following settings are used to configure the PNA for remote control using SCPI commands.

How to Configure for SICL / GPIB Operation

Using front-panel

HARDKEY [softkey] buttons

1. Press **SYSTEM**
2. then **[Configure]**
3. then **[SICL/GPIB]**

PNA Menu using a mouse

1. Click **Utility**
2. then **System**
3. then **Configure**
4. then **SICL/GPIB/SCPI**

Programming Commands

SICL / GPIB dialog box help

The screenshot shows a dialog box titled "SICL / GPIB / SCPI" with the following sections and options:

- GPIB**
 - GPIB(1) Talker/Listener Address: 16
- SICL**
 - SICL Enabled (Standard Instrument Control Library)
 - Address: 16
 - Automatically Enable on Startup
- LAN Sockets/Telnet**
 - Sockets Enabled Telnet Enabled
- HiSlip**
 - HiSlip Enabled Address: 0
- SCPI Monitor/Input**
 - GPIB Command Processor Console
 - Monitor GPIB Bus
 - Show GPIB Bus Monitor Window

Buttons: OK, Cancel, Help

GPIB

Talker/Listener Address Sets the PNA address used to send and receive GPIB/SCPI commands to the system controller (external computer).

Use the National Instruments interface or the ACE (Agilent Connection Expert) interface to change the System Controller address. Use the PNA as the system controller of external devices. Learn about the [PNA as controller](#).

See the rear panel of the [PNA-X](#) and N522x models.

SICL

SICL Enabled When checked, the analyzer is capable of running GPIB programs on its computer to control analyzer functions. The programs must be run from a GPIB-capable programming environment (VEE, Visual Basic). This mode does not allow control of external GPIB instruments. To uncheck this box, exit the PNA application - (Click File, then Exit). The PNA restarts with the SICL enabled box unchecked unless **Automatically Enable on Startup** is checked.

Learn more about [Configuring for VISA and SICL](#).

Note: When **SICL Enabled** is checked, the PNA VXI-11.2 interface is enabled, and if the PNA hard disk image is new enough to have the VXI-11.3 interface, it also enables that. [Learn more about LXI/VXI](#).

Address Sets the PNA address.

Automatically Enable on Startup When checked, SICL Enabled is automatically selected when starting the PNA application.

LAN Sockets/Telnet

Provides ability to communicate with the PNA from a PC that uses a Windows, or non-Windows, operating system.

- These settings are checked by default. If you have security concerns, clear these check boxes.
- These settings remain after the PNA is shutdown and restarted.

Sockets Enabled When checked, provides the ability to control the PNA from a remote SCPI program using port number 5025. [See the C# example that illustrates how this is done](#).

Telnet Enabled When checked, provides the ability to send single SCPI commands from a remote Windows, or non-Windows, PC to the PNA using port number 5024.

How to send single SCPI commands using Telnet:

1. On the remote PC, click **Start**, then **Run**
2. Type: **telnet** <computer name> **5024**
where <computer name> is the full computer name of the PNA. [See how to find the computer name of the PNA.](#)
3. A Telnet window with a **SCPI>** prompt should appear on the remote PC screen.
4. From the SCPI prompt:
 - Type single SCPI commands
 - If an invalid SCPI command is sent, the prompt will disappear. Press **Enter** or **Ctrl C** to recover the SCPI prompt.
 - To exit the telnet window click **X** in the upper-right corner.
 - To get a normal telnet prompt, press **Ctrl]** (closing bracket).
 - To close the normal telnet window, type **Quit** and press **Enter**.

HiSLIP

HiSLIP has the same functionality as VXI 11 (SICL) with better performance. Therefore, it is enabled by default on the PNA.

Address 0 by default. No change is necessary for the PNA.

On the remote computer, use address string TCPIP0::::hislip<address>. ("hislip" is case-sensitive).

SCPI Monitor / Input

GPIB Command Processor Console Launches a window that is used to send single SCPI/GPIB commands from the PNA keyboard.

Note: Press **Control+Z** , then enter, to close the console window.

Note: The Status Register system can NOT be used from the GPIB Console.

- Type a valid command, with appropriate arguments and press enter.
- Use the arrow keys to recall previous commands.
- The console window may launch behind the PNA application. Press **Control+Tab** to bring

the console window to the top.

Monitor GPIB Bus Enables monitoring activity on the GPIB.

Show GPIB Bus Monitor Window Shows and hides the window monitoring GPIB activity.

Local and Remote Operation

The analyzer **LCL** and **RMT** (Local and Remote) operation labels appear in the lower right corner of the status bar.

Note: The status bar is NOT visible when the analyzer is preset. See [how to make the status bar visible](#).

- **LCL** appears when NOT under SCPI control
- **RMT** appears when under SCPI control. The RMT label does NOT appear when under COM control. Remote operation disables the front panel keys except for the **Macro/Local** key.

To return to Local (front panel) operation, press the Macro / Local key

Sending the GPIB "GTL" (go to local) command also returns the analyzer to Local operation.

Sending the GPIB "LLO" (local lockout) command disables the front panel Local button.

Last Modified:

1-May-2009	Added note about Status Register and Console
30-Oct-2008	Removed legacy content.
21-Feb-2008	Include Windows OS in Telnet/Sockets

Getting Data from the Analyzer

Data is sent from the analyzer in response to program queries. Data can be short response messages, such as analyzer settings, or large blocks of measurement data. This topic discusses how to read query responses and measurement data from the analyzer in the most efficient manner.

- [Response Message Syntax](#)
- [Clearing the Output Queue](#)
- [Response Data Types](#)
- [Transferring Measurement Data](#)

Note: Some PCs use a modification of the IEEE floating point formats with the byte order reversed. To reverse the byte order for data transfer into a PC, use the [FORMat:BOReR](#) command.

Other Topics about GPIB Concepts

Response Message Syntax

Responses sent from the analyzer contain data, appropriate punctuation, and message terminators.

<NL><^END> is always sent as a response message terminator. Most programming languages handle these terminators transparent to the programmer.

Response messages use commas and semicolons as separators in the following situations:

- a comma separates response data items when a single query command returns multiple values

```
FORM:DATA? 'Query  
ASC, +0 'Analyzer Response
```

- a semicolon separates response data when multiple queries are sent within the same messages

```
SENS:FREQ:STAR?;STOP? --Example Query
```

```
+1.23000000000E+008; +7.89000000000E+008<NL><^END> 'Analyzer Response
```

Clearing the Output Queue

After receiving a query, the analyzer places the response message in its output queue. Your program should read the response immediately after the query is sent. This ensures that the response is not cleared before it is read. The response is cleared when one of the following conditions occur:

- When the query is not properly terminated with an ASCII carriage return character or the GPIB <^END> message.
- When a second program query is sent.
- When a program message is sent that exceeds the length of the input queue
- When a response message generates more response data than fits in the output queue.
- When the analyzer is switched ON.

Response Data Types

The analyzer sends different response data types depending on the parameter being queried. You need to know the type of data that will be returned so that you can declare the appropriate type of variable to accept the data. For more information on declaring variables see your programming language manual. The GPIB Command Finder lists every GPIB command and the return format of data in response to a query. The analyzer returns the following types of data:

[Numeric Data](#)

[Character Data](#)

[String Data](#)

[Block Data](#)

Numeric Data

All numeric data sent over the GPIB is ASCII character data. Your programming environment may convert the character data to numeric data for you. Boolean data (1 | 0) is a type of numeric data.

Character Data

Character data consists of ASCII characters grouped together in mnemonics that represent specific analyzer settings. The analyzer always returns the short form of the mnemonic in upper-case alpha characters. Character data looks like string data. Therefore, refer to the GPIB Command Finder to determine the return format for every command that can be queried.

Example of Character Data

```
MLOG
```

String Data

String data consists of ASCII characters. String parameters can contain virtually any set of ASCII characters. When sending string data to the analyzer, the string **must** begin with a single quote (') or a

double quote (") and end with the same character (called the delimiter).

Note: The analyzer responds best to all special characters if the string is enclosed in single quotes. If quotes are not used, the analyzer will convert the text to uppercase. The analyzer may not respond as you expect.

The analyzer always encloses data in double quotes when it returns string data.

Example of String Data

```
 GPIB.Write "DISP:WINDow:TITLe:DATA?"
```

```
"This is string response data."
```

Block Data

Block data is used to transfer measurement data. Although the analyzer will accept either definite length blocks or indefinite length blocks, it always returns definite length block data in response to queries unless the specified format is ASCII. The following graphic shows the syntax for definite block data:



<num_digits> specifies how many digits are contained in <byte_count>

<byte_count> specifies how many data bytes will follow in <data bytes>

Example of Definite Block Data

```
#210ABCDE+WXYZ<nl><end>
```

Where:

- always sent before definite block data

2 - specifies that the byte count is two digits (2)

10 - specifies the number of data bytes that will follow, not counting <NL><END>

ABCDE+WXYZ - 10 digits of data

<NL><END> - always sent at the end of block data

Transferring Measurement Data

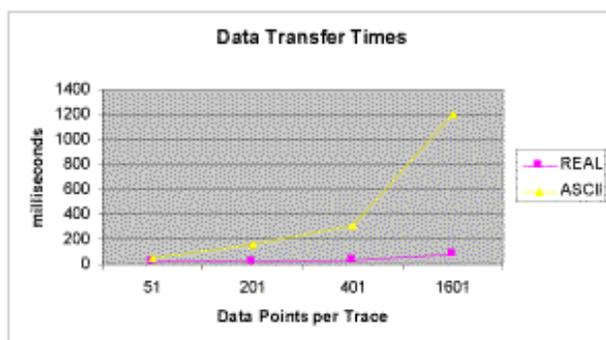
Measurement data is blocks of numbers that result from an analyzer measurement. Measurement data is available from various processing arrays within the analyzer. For more information on the analyzer's data processing flow, see [Accessing Data Map](#). Regardless of which measurement array is read, transferring measurement data is done the same.

[See an example.](#)

When transferring measurement data, the [FORMat:DATA](#) command allows you to choose from the following two data types:

- REAL
- ASCII

The following graphic shows the differences in transfer times between the two:



REAL Data

REAL data (also called floating-point data) types transfer faster. This is because REAL data is binary and takes about half the space of ASCII data. The disadvantage of using REAL data is that it requires a header that must be read. See [definite length block data](#). The binary floating-point formats are defined in the IEEE 754-1985 standard. The following choices are available in REAL format:

- **REAL,32** - IEEE 32-bit format - single precision (not supported by HP BASIC)
- **REAL,64** - IEEE 64-bit format - double precision

ASCII Data

The easiest and slowest way to transfer measurement data is to use ASCII data. ASCII data is sent if the data contains both numbers and characters (the setting of FORMat:DATA is ignored). ASCII data is separated by commas.

6-Feb-2013	Changed block data example
13-Jul-2012	Fixed typo
28-Jul-2009	Added three zeros to example
26-Jul-2007	Added link to example

Synchronizing the PNA and Controller

Synchronizing the PNA and Controller means to keep PNA and the controller working at approximately the same pace. In this topic:

- [The Problem and the Solution](#)
- [PNA Queues](#)
- [Synchronization Methods](#)
- [When To Synchronize the Analyzer and Controller](#)
 - [Completion of a Measurement](#)
 - [Measurements with External Trigger](#)
 - [Averaged Measurements](#)
 - [During Calibration Acquire](#)

See Also

- [Synchronize an External PSG Source](#)
- [Triggering the PNA using SCPI](#)

The Problem

The controller sends commands to the PNA as fast as the GPIB bus will allow. The PNA stores these commands in the PNA [Input queue](#). However, the PNA executes those commands at a slower rate than they are accepted. If left unchecked, the PNA input buffer will contain a long list of commands waiting to be executed.

At some point, the controller will send a query command which requires a response from the PNA. The controller will not send more commands until a response is received. It will wait for a response from the PNA for the amount of time set by the Timeout setting. If the PNA is working off a long list of commands in the input buffer, it may not execute and respond to the query command until the controller has quit waiting, or "timed out".

The Solution

The easiest way to keep the controller and the PNA "synched" is to send query commands often. This stops the controller from sending more commands until the PNA executes and responds to the query. This limits the number of commands that are waiting in the PNA input queue to be processed.

Although any query will stop the controller from sending more commands, a good practice is to send [*OPC?](#) Most of the time, as soon as this query is executed, the PNA will immediately reply. The exception to this is the Overlapped command.

- **Sequential** commands are executed quickly and in the order in which they are received.
- **Overlapped** (also known as Asynchronous) commands take longer to execute. Therefore, they allow the PNA to execute other commands while waiting. However, the programmer may want to prevent the analyzer from processing new commands until the overlapped command has completed. If the PNA is executing an overlapped command when a [*OPC?](#) is received, it will wait until the overlapped command is complete before processing new commands.

Note: The analyzer has two overlapped commands:

- [INITiate:IMMediate](#)
- [SENSe:SWEep:MODE GROUPS](#) (when INIT:CONT is ON)

Several calibration commands have an optional ASYNchronous argument which allows them to behave like overlapped commands. [Learn more.](#)

Analyzer Queues

Queues are memory buffers that store messages until they can be processed. The analyzer has the following queues:

- [Input Queue](#)
- [Output Queue](#)
- [Error Queue](#)

Input Queue

The controller sends statements to the analyzer without regard to the amount of time required to execute the statements. The input queue is very large (31k bytes). It temporarily stores commands and queries from the controller until they are read by the analyzer's command parser. The input queue is cleared when the analyzer is switched ON.

Output Queue

When the analyzer parses a query, the response is placed in the output queue until the controller reads it. Your program should immediately read the response or it may be cleared from the output queue. The following conditions will clear a query response:

- When a second query is sent before reading the response to the first. This does not apply when multiple

queries are sent in the same statement.

- When a program statement is sent that exceeds the length of the input queue.
- When a response statement generates more data than fits in the output queue.
- When the analyzer is switched ON.

Error Queue

Each time the analyzer detects an error, it places a message in the error queue. When the `SYSTEM:ERROR?` query is sent, one message is moved from the error queue to the output queue so it can be read by the controller. Error messages are delivered to the output queue in the order they were received. The error queue is cleared when any of the following conditions occur:

- When the analyzer is switched ON.
- When the `*CLS` command is sent to the analyzer.
- When all of the errors are read.

If the error queue overflows, the last error is replaced with a "Queue Overflow" error. The oldest errors remain in the queue and the most recent error is discarded.

Synchronization Methods

The following common commands are used to synchronize the analyzer and controller. Examples are included that illustrate the use of each command in a program. See the SCPI command details to determine if a command is an overlapped command.

- [*WAI](#)
- [*OPC?](#)
- [*OPC](#)

*WAI

The `*WAI` command:

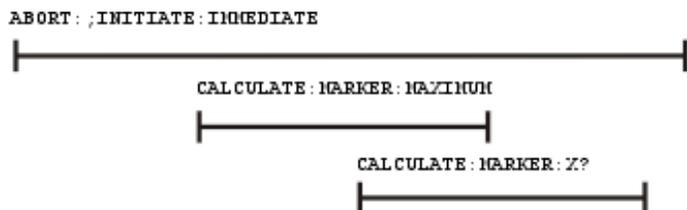
- **Stops the analyzer** from processing subsequent commands until all overlapped commands are completed.
- **It does NOT stop the controller** from sending commands to this and other devices on the bus. This is the easiest method of synchronization.

Example of the *WAI command

```
GPIB.Write "ABORT;:INITIATE:IMMEDIATE" 'Restart the measurement.'
```

```
GPIB.Write "CALCULATE:MARKER:SEARCH:MAXIMUM" 'Search for max amplitude.  
GPIB.Write "CALCULATE:MARKER:X?" 'Which frequency?
```

The following time line shows how the processing times of the three commands relate to each other:

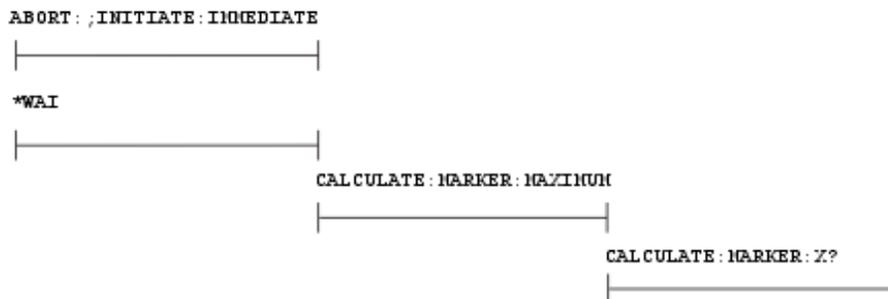


`INITIATE:IMMEDIATE` is an overlapped command. It allows the immediate processing of the sequential command, `CALCULATE:MARKER:SEARCH:MAXIMUM`. However, the `INITIATE:IMMEDIATE` is not considered complete until the measurement is complete. Therefore, the marker searches for maximum amplitude before the measurement completes. **The `CALCULATE:MARKER:X?` query could return an inaccurate value.**

To solve the problem, insert a `*WAI` command.

```
GPIB.Write "ABORT;:INITIATE:IMMEDIATE" 'Restart the measurement.  
GPIB.Write "*WAI" 'Wait until complete.  
GPIB.Write "CALCULATE:MARKER:MAXIMUM" 'Search for max amplitude.  
GPIB.Write "CALCULATE:MARKER:X?" 'Which frequency
```

The time line now looks like this:



The `*WAI` command keeps the `MARKER:SEARCH:MAXIMUM` from taking place until the measurement is completed. The `CALCULATE:MARKER:X?` query returns the correct value.

Note: Although `*WAI` stops the analyzer from processing subsequent commands, it does not stop the controller. The controller could send commands to other devices on the bus.

*OPC?

The `*OPC?` query stops the controller until all pending commands are completed.

In the following example, the **Read** statement following the *OPC? query will not complete until the analyzer responds, which will not happen until all pending commands have finished. Therefore, the analyzer and other devices receive no subsequent commands. A "1" is placed in the analyzer output queue when the analyzer completes processing an overlapped command. The "1" in the output queue satisfies the **Read** command and the program continues.

Example of the *OPC? query

This program determines which frequency contains the maximum amplitude.

```
GPIB.Write "ABORT; :INITIATE:IMMEDIATE"! Restart the measurement
GPIB.Write "*OPC?" 'Wait until complete
Meas_done = GPIB.Read 'Read output queue, throw away result
GPIB.Write "CALCULATE:MARKER:MAX" 'Search for max amplitude
GPIB.Write "CALCULATE:MARKER:X?" 'Which frequency?
Marker_x = GPIB.Read
PRINT "MARKER at " & Marker_x & " Hz"
```

*OPC

The *OPC command allows the analyzer and the controller to process commands while processing the overlapped command.

When the analyzer completes processing an overlapped command, the *OPC command sets bit 0 of the standard event register to 1. This requires polling of status bytes or use of the service request (SRQ) capabilities of your controller. See [Reading the Analyzer's Status Registers](#) for more information about the standard event status register, generating SRQs, and handling interrupts.

Note: Be careful when sending commands to the analyzer between the time you send *OPC and the time you receive the interrupt. Some commands could jeopardize the integrity of your measurement. It also could affect how the instrument responds to the previously sent *OPC.

Example of polled bit and SRQ processes.

When To Synchronize the Analyzer and Controller

The need to synchronize depends upon the situation in which the overlapped command is executed. The following section describes situations when synchronization is required to ensure a successful operation.

- [Completion of a Measurement](#)
- [Measurements with External Trigger](#)
- [Averaged Measurements](#)

Completion of a Measurement

To synchronize the analyzer and controller to the completion of a measurement, use the `ABORT ; INITIATE : IMMEDIATE` command sequence to initiate the measurement.

This command sequence forces data collection to start (or restart) under the current measurement configuration. A restart sequence, such as `ABORT ; INITIATE : IMMEDIATE` is an overlapped command. It is complete when all operations initiated by that restart command sequence, including the measurement, are finished. The `*WAI`, `*OPC?` and `*OPC` commands allow you to determine when a measurement is complete. This ensures that valid measurement data is available for further processing.

Measurements with External Trigger

See [Triggering the PNA using SCPI](#).

[External Triggering](#)

Averaged Measurements

Averaged measurements are complete when the average count is reached. The average count is reached when the specified number of individual measurements is combined into one averaged measurement result. Use synchronization to determine when the average count has been reached.

If the analyzer continues to measure and average the results after the average count is reached, use synchronization to determine when each subsequent measurement is complete.

During Calibration Acquire

During a calibration with slow sweep speeds, such as when using a narrow IF bandwidth, you may want to have your program perform other operations, such as checking for the click event of a Cancel button.

To do this, use the optional `ASYNchronous` argument with the `ACQUIRE` command as shown in several calibration example programs. The PNA parser returns immediately while the cal step measurement proceeds. It does NOT block commands and wait for the measurement step to finish. You can send `*ESR?` or `*STB?` queries to monitor the status register bytes to see when the OPC (operation complete) bit gets set, which indicates the cal measurement step has finished. Learn more about [status registers](#).

Note: Do NOT issue the `*OPC?` command when using the `ASYN` argument. If your program is using the [ScpiStringParser](#), then you can ONLY use `*OPC?` to detect when the OPC bit is set, so do NOT use the `ASYN` argument with the calibration commands when using that parser.

When using the `ASYN` argument, set the timeout value in the IO settings to at least 5 seconds. There are intervals during the cal acquires when the PNA takes a several seconds to respond to additional commands, such as when the processor is calculating error terms.

The following commands have this argument:

Command	Example
SENS:CORR:COLL:GUID:ACQuire (Guided Cal)	Guided 2-Port or 4-Port Cal
SENS:CORR:COLL:ACQuire (Unguided Cal)	Perform Unguided ECAL
SOUR:POW:CORR:COLL:ACQuire (Source Power Cal)	Perform a Source and Receiver Power Cal (shows polling loop)

In addition, the [SENS:CORR:COLL:GUIDed:INITialize](#) command has this optional argument for long calibration initialization, such as a [CalAll](#) calibration.

Last Modified:

27-Sep-2012 Added Cal Acq info
6-Apr-2009 Added links to External trig and triggering
17-Feb-2009 Added Cal ACQ to Overlapped commands

Calibrating the PNA Using SCPI

There are several ways to calibrate the PNA using SCPI depending on your measurement needs. As from the Cal Wizard, you can perform a Guided Cal, Unguided Cal, or ECal. This topic explains the differences in these calibration choices when using SCPI commands.

- [Guided Calibrations](#)
- [ECal](#)
- [Creating Cal Sets](#)
- [Applying Cal Sets and Cal Types](#)
- [Uploading Error Terms](#)
- [Unguided Cals and Calibration Classes](#)

See Also

[Synchronizing the PNA and Controller \(During a calibration\)](#)

[See SCPI Calibration Examples](#)

Guided Calibrations

Guided versus Unguided is the style of calibration that is selected on the first page of the [Calibration Wizard](#). A remote 'guided' cal does not present the cal wizard, but prompts for specific standards to be connected. In a remote 'Unguided', the steps must be 'hard-coded'.

- To perform a **Guided Calibration**, use ONLY [Sens:Corr:Coll:Guided](#) commands.
- These commands calibrate the ACTIVE channel. Activate a channel by selecting a measurement on the channel to be calibrated using [Calc:Par>Select](#).
- Full 1,2,3,4-port SOLT and TRL calibrations - No response cals.
- All of the advanced calibration features (Thru method, specify DUT connectors and Cal kits for each port, port pairings).
- A Cal Set is applied to the channel and saved at the completion of a guided cal according to the preference setting [SENS:CORR:PREF:CSET:SAVE](#)

Note: To perform an **Unguided Calibration**, use ONLY the [Sens:Corr](#) commands (NOT Guided).

ECal

From the Cal Wizard or from a SCPI program, ECal is fast, accurate, and very repeatable. Unlike from the Cal Wizard, you can use SCPI to perform ECal using either the Guided or Unguided commands. The Unguided commands are easiest to use. However, the following situations require that you use the Guided commands.

- To maximize accuracy, all ECal calibrations on the PNA perform an Unknown Thru measurement of the ECal module Thru state **IF** the PNA model being used has [1 reference receiver per port](#). If your PNA does NOT have 1 reference receiver per port, use Guided ECal commands and specify a Thru method.
- If your ECal module connectors do NOT match the DUT connectors, and you choose not to perform a User Characterization, use Guided ECal commands and specify the Thru method.

ECAL Notes:

- When using either Guided or Unguided ECal commands under low power situations, use the Orientation settings. The Guided example shows the use of these commands. When using Unguided, they must appear before the Acquire command.
- The frequency range of the measurement must be within the range of the ECal module. Otherwise, the calibration will fail.
- You do NOT have to send the ECal module state 'switch' commands. The ECal algorithm switches ECal states automatically.
- All of these ECal choices are listed in the [Programming Command Finder](#) function in this Help file.

See [Using ECal](#) to learn about all of the ECal features.

Creating Cal Sets

There are several ways to store guided cal data into a unique Cal Set. The following is probably the easiest. It does not require the name of an existing Cal Set and it allows you to name the Cal Set.

```
SENS:CORR:COLL:GUID:INIT 'start the cal with no cal set argument  
  
'Perform the cal  
  
SENS:CORR:COLL:GUID:SAVE 'create cal set with auto-generated name or to cal  
register  
  
SENS:CORR:CSET:NAME 'MyCalSet' 'name the current cal set.
```

Applying Cal Sets and Cal Types

A Cal Set is applied to the channel and saved at the completion of a guided cal according to the

preference setting [SENS:CORR:PREF:CSET:SAVE](#).

When you select a Cal Set to apply to an uncalibrated channel, the PNA attempts to find the most comprehensive calibration type in the Cal Set and turn it ON. In addition, changing a measurement parameter (for example, from S11 to S21) will also initiate an attempt to apply the best Cal Type and turn correction ON.

There may be times when you do not want the most comprehensive Cal Type. For example, say there is a Full 2-port Cal Set applied, but there is only an S11 measurement displayed. If measurement speed is a concern, you can apply a Full 1-Port Cal Type from that same Cal Set and save time by not doing the extra background sweeps. [Learn more about background sweeps](#).

If you change the measurement parameter, the PNA will reapply the Full 2-Port Cal Type.

See the SCPI and COM commands for [Cal Sets](#) and [Cal Types](#).

Uploading Error Terms

Note: There was a method described here for WinCal 3.x that involved a [preference setting](#). That method is no longer supported.

To upload error terms into a created or selected Cal Set:

```
SENS:CORR:CSET:CREATE or SENS:CORR:CSET:GUID
SENS:CORR:CSET:Data <term> <port> <port> <data>
SENS:CORR:CSET:SAVE
```

This method puts error terms into a Cal Set, outside of a Guided or Unguided calibration session.

The Cal Set can then be applied at any time.

See [SENS:CORR:CSET](#) commands.

Unguided Cals and Calibration Classes

- Use [Sens:Correction](#) commands.
- 1-port, 2-port, Response.
- Can select 2 sets of standards.
- TRL is NOT recommended.

The following describes how to perform an unguided calibration using SCPI. The objective here is to make clear the relationship between the physical port on which a standard is being measured, the actual device in the cal kit, and the SCPI command used to acquire the device.

Calibration standards classes are 'categories' of standard types. To perform a 2 port calibration, the cal wizard requires the following types of standards to be measured:

3 reflection standards on the forward port:

- Class S11A typically an open
- Class S11B typically a short
- Class S11C typically a load

Likewise, 3 reflection standards are required for the reverse port:

- Class S22A typically an open
- Class S22B typically a short
- Class S22C typically a load

There is also a transmission standard that is measured in both directions:

- Class S21T typically a thru

The following illustrates the relationship between cal kit physical standards and calibration classes. Here is a list of the physical devices in my calibration kit.

Standard #1 = "3.5 mm male short"

Standard #2 = "3.5 mm male open"

Standard #3 = "3.5 mm male broadband load"

Standard #4 = "Insertable thru standard"

Standard #5 = "3.5 mm male sliding load"

Standard #6 = "3.5 mm male lowband load"

Standard #7 = "3.5 mm female short"

Standard #8 = "female to female characterized thru adapter"

Standard #9 = "0-2 Load"

Standard #10 = "Open"

Standard #11 = "Non-insertable thru"

Standard #12 = "3.5 mm female lowband load"

Standard #13 = "3.5 mm female sliding load"

Standard #14 = "3.5 mm female broadband load"

Standard #15 = "3.5 mm female open"

When you perform a calibration remotely using SCPI, you don't specify the device number directly.

Rather, you specify the class you want to measure. Each device in the calibration kit is assigned to a class. And since more than one device can be assigned to the same class, each class contains an ordered list of devices. The class assignments are set using the Advanced Modify Cal Kit dialog or the SCPI command:

[SENS:CORR:COLL:CKIT:ORDer](#)<class>, <std>, <std>, <std>, <std>,<std>,<std>,<std>

The 85052B kit used in the example program has the following standard list for each class: The list was obtained by issuing the corresponding SCPI query:

[SENS:CORR:COLL:CKIT:OLIST1?](#) S11A = +2,+15,+0,+0,+0,+0,+0

SENS:CORR:COLL:CKIT:OLIST2? S11B = +1,+7,+0,+0,+0,+0,+0

SENS:CORR:COLL:CKIT:OLIST3? S11C = +6,+5,+3,+12,+13,+14,+0

SENS:CORR:COLL:CKIT:OLIST4? S21T = +4,+8,+0,+0,+0,+0,+0

SENS:CORR:COLL:CKIT:OLIST5? S22A = +2,+15,+0,+0,+0,+0,+0

SENS:CORR:COLL:CKIT:OLIST6? S22B = +1,+7,+0,+0,+0,+0,+0

SENS:CORR:COLL:CKIT:OLIST7? S22C = +6,+5,+3,+12,+13,+14,+0

SENS:CORR:COLL:CKIT:OLIST8? S12T = +4,+8,+0,+0,+0,+0,+0

When you perform the calibration, you acquire data by issuing the ACQUIRE command:

[SENS:CORR:COLL:ACQ](#) <class>[, <subst>]

For example:

[SENS:CORR:COLL:SFOR](#) 1

SENS:CORR:COLL:ACQ STANA, SST2

The SFOR command tells the wizard to make the next acquisition in the forward direction. The ACQUIRE command specifies that we are measuring the 2nd device in the list for STANA. And since

we are measuring SFORward, then STANA refers to class #1 or S11A. The list of devices for this class are specified in the OLIST1 query above.

Alternately, you could modify the device order for the S11A class to move device #15 into the first position (SENS:CORR:COLL:CKIT:ORDER1). When the desired device is in the first position, you need not specify the order number in the ACQUIRE command. The default is the first device in the OLIST. This works well for two port network analyzers where the order for S11A,B,C classes is set up for port 1 and the order for S22A,B,C is set up for port 2. With the kit set up in the proper order, you eliminate the need to specify the substandard number (SST<n>).

[See an example: Perform an Unguided 2-port Cal on a 4-port PNA.](#)

Last Modified:

7-Nov-2012	Added intro
17-Apr-2009	Remove *OPC? from Sync note
13-Feb-2009	Added Asynch section

The PNA as a USB Device

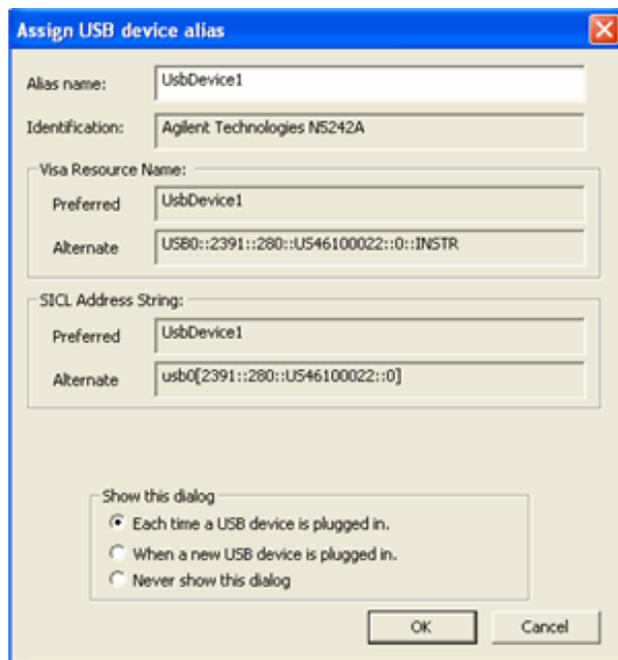
Beginning with PNA Rev. A.09.00, the PNA can be controlled as a USB Device using SCPI. This is done through the Agilent I/O Libraries which must be installed on your remote computer.

All data types, especially [Binary block data](#), transfer MUCH faster using USB as compared to GPIB.

To communicate with the PNA as a USB device

1. Connect the PNA to the remote computer using the rear-panel [device-side USB connector](#).
2. The 'Found New Hardware' wizard is launched. Follow the prompts to install the PNA driver software.
3. The Agilent I/O Libraries will recognize the PNA as a Test and Measurement device and show the following dialog.

Note: The PNA is not a USB Mass Storage Device. Therefore, Windows Explorer does NOT recognize it as a USB device. You can NOT use Windows Explorer to transfer files to and from the PNA. For file transfer, use the SCPI command [MMEM:TRANsfer](#).



Alias name Change this to a name that is easy to recognize. Once configured, use the Alias name to communicate with the USB device using applications such as VISA and SICL:

- VISA: `viOpen (...,"UsbDevice1",...)`
- SICL: `iopen ("UsbDevice1")`

For more information, see the Connectivity Guide in the Agilent I/O libraries.

Last Modified:

9-Apr-2012	Removed links to old PNA
26-Jul-2007	MX New topic

Reading the Analyzer's Status Register

The PNA has several status registers that your program can read to know when specific events occur. There are two methods of reading the status registers in the analyzer: the Polled Bit method and the Service Request method.

- [The Status Registers](#)
- [Setting and Reading Bits in Status Registers](#)
- [Polled Bit Method](#)
- [Service Request Method](#)

See Also

[IEE 482 Common commands](#)

[Example: Status Reporting](#)

[Status Commands](#)

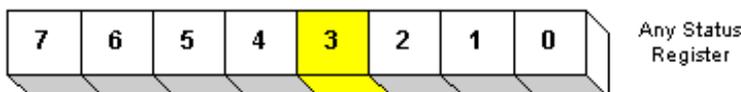
Other Topics about GPIB Concepts

Important Notes:

- A new [Limit Line Fail command](#) that makes it easy to determine if Limit Line testing has failed.
- [*OPC?](#) can be used to easily determine when a channel has completed a sweep. This requires no interaction with the Status Register system. Most [PNA programming examples](#) use *OPC?.
- Most of the Status Register system can NOT be used with the [SCPIStringParser Object](#). However, *OPC? can be used.

The Status Registers

Most of the status registers in the analyzer have sixteen bits. For simplicity, this topic will illustrate their use with 8-bit registers. Bits in registers represent the status of different conditions inside of the analyzer. In the following graphic, a register is represented by a row of boxes; each box represents a bit. Bit 3 is ON.



Each PNA Status Register is actually comprised of the following registers. [See an image of the PNA](#)

Status registers.

- **Enable Registers** - When using the [SRQ method of polling](#), you first set bits in the enable register which tells the PNA which events to monitor. This is not necessary using the [Polled Bit method](#), as you can only monitor a single event. A *CLS (clear status) command will not clear the enable register. The *ESE and *ESE? commands are used to set and query Enable bits, while *ESR is used to read and clear an Enable register. [Learn how to set bits.](#)
- **Condition Registers** - A condition register continuously monitors events in the PNA. Bits in the condition register change real time as conditions occur. These bits are not latched, so this register is used mainly for diagnostic purposes. The registers that only summarize lower level registers do NOT have a condition register.
- **Event Registers** - This is the register that is read to determine if an event has occurred. An event register latches the bits from the corresponding condition register. When an event register bit is set, subsequent changes to the corresponding condition register bit are ignored. The bit remains set until a query command such as *CLS clears the bit. [Learn how to read the Event Register.](#)
- **Positive and Negative Transition Registers** - Transition registers control what type of in a condition register will set the corresponding bit in the event register.
 - **Positive** transitions (**0 to 1**) are only reported to the event register if the corresponding positive transition bit is set to 1.
 - **Negative** transitions (**1 to 0**) are only reported to the event register if the corresponding negative transition bit is set to 1.
 - Setting **both** transition bits to 1 causes both **positive and negative** transitions to be reported.

Transition registers are read-write and are unaffected by *CLS (clear status) or queries. They are reset to their default settings at power-up and after *RST and SYSTem:PRESet commands. The **following are the default settings** for the transition registers:

- All Positive Transition registers = 1
- All Negative Transition registers = 0

This means that, by default, the analyzer will latch all event registers on the negative to positive transition (0 to 1).

The following is an example of why you would set transition registers:

A critical measurement requires that you average 10 measurements and then restart averaging. You decide to poll the averaging bit. When averaging is complete, the bit makes a positive transition. After restart, you poll the bit to ensure that it is set back from 1 to 0, a negative transition. You set the negative transition bit for the averaging register.

Setting and Reading Bits in Status Registers

Both the Polled-Bit method and Service Request method require that you set and read status register bits. Most of the PNA status registers contain 16 bits, numbered 0 to 15. Each bit has a weighted value. The following example shows how to set the bits in a 8-bit status register.

8-bit register

Bit	0	1	2	3	4	5	6	7
Weight	1	2	4	8	16	32	64	128

How to set bits 4 and 5 in the Standard Event Status Enable register:

Step	Example
1. Determine the weighted bit value for these weights 16 and 32 (respectively) bits	
2. Add these values together	$16 + 32 = 48$
3. Send this number as an argument in the appropriate command. (see Status Commands)	STAT:QUES:LIMIT1:ENAB 48

The Polled Bit Method

With the Polled Bit Method, your program monitors a bit in the status register that represents the condition of interest to you. When the PNA sets the bit to 1, your program sees it and responds accordingly.

- If your program **periodically** monitors a bit in the status register, it is free to do other things as well. However, your program can respond only as fast as the bit is polled.
- If your program **continually** monitors a bit, it can respond immediately, but will be unavailable to do anything other than poll the bit.

Advantage: This method requires very little programming.

Procedure:

1. Decide which condition to monitor. The [Status Commands](#) topic lists all of the possible conditions that can be monitored in the analyzer.
 2. Determine the command to be used to monitor the bit.
 3. Determine how often to poll the bit until it is set.
 4. Construct the routine to respond when the bit is set.
-

The Service Request (SRQ) Method

Your program enables the bits in the status registers representing the condition of interest. When the condition occurs, the PNA actively interrupts your program from whatever it is doing, and an event handler in your program responds accordingly. Do this method if you have several conditions you want to monitor or the conditions are such that it is not practical to wait for the condition to occur.

Advantage: This method frees your program to do other things until the condition occurs. The program is interrupted to respond to the condition.

Disadvantage: This method can require extensive programming depending on the number and type of conditions that you want to monitor.

Procedure:

1. Decide which conditions to monitor. The [Status Commands](#) topic lists all of the possible analyzer conditions that can be monitored.
2. Set the enable bits in the **summary** registers and the **status byte** register.

Enabling is like making power available to a light. Without power available, the switch can be activated, but the light won't turn ON. In the analyzer, without first enabling a bit, the condition may occur, but the controller won't see it unless it is enabled.

The condition, and the bit in the **summary** registers in the reporting path, must be enabled. This is like streams (conditions) flowing into rivers (summary registers), and rivers flowing into the ocean (controller). See the diagram of status registers in [Status Commands](#).

Bit 6 of the **status byte** register is the only bit that can interrupt the controller. When **any** representative bit in the status byte register goes ON, bit 6 is automatically switched ON.

3. Enable your program to interrupt the controller. This is done several ways depending on the programming language and GPIB interface card you use. An [example program](#) is provided showing how this is done with in Visual Basic with a National Instruments GPIB card.
4. Construct a subroutine to handle the interrupt event. If you are monitoring more than one condition in your system, your event handler must determine which condition caused the interrupt. Use the *SPE command to determine the instrument that caused the interrupt, then poll the summary registers, then poll condition registers to determine the cause of the interrupt.

Last Modified:

4-Jun-2009	Modified Polled Bit method
1-May-2009	Several edits

Referring to Traces, Measurements, Channels, and Windows Using SCPI

Sometimes in a SCPI program you may need to refer to traces that you have not created. This can be a bit confusing in the PNA. Here are the THREE ways to refer to a specific measurement trace.

Note: The terms "Trace" and "Measurement" effectively mean the same thing in this discussion.

1. The **Measurement Name** is picked by you when you first create a trace using the [CALCulate<cnum>:PARAmeter\[:DEFine\]:EXTended <Mname>,<param>](#) command. The measurement name is only used by SCPI.
2. The **Trace Number** is also picked by you when 'feeding' a newly-created measurement name to a window number using [DISP:WINDow<wmun>:TRACe<tnum>:FEED](#). The trace number is used ONLY by SCPI and is mainly used to refer to traces in the DISPlay node. This is NOT the number that appears as **Tr#** on the screen. While you can assign any Trace number you want, when a measurement is created from the GUI, the PNA assigns numbers to the traces sequentially, starting with one in each window. Therefore, when there is more than one window, these numbers are not unique.
3. The **Tr#** that appears on the PNA screen is the third and most visible way to refer to a trace. Since we already have a "Trace Number", we call this the **Measurement Number** in the PNA Help file. This number is issued sequentially by the PNA regardless of channel and window. It is therefore unique among all traces. This command returns the name of the active measurement: [SYSTem:ACTive:MEAS?](#)

The concept of the **Active measurement** versus **Selected Measurement** is also a bit confusing. As seen on the screen, the Active measurement has the highlighted Tr# . While there can only be ONE active measurement, every channel has a selected measurement. The target measurement must first be selected before most CALC node settings can be made. There are two ways to select a measurement for each channel:

1. Use [CALC<ch>:PAR:SEL <measName>](#) which requires the channel number and measurement name.
2. Use [CALC<ch>:PAR:MNUM <measNum>](#) which requires the channel and measurement (**Tr**) number.

Here are other relevant commands for referring to traces, measurements, channels, and windows:

- [CALC<cnum>:PAR:CATalog:EXTended?](#) - Catalog the Measurement Names for the specified channel.
- [CALC<cnum>:PAR:TNUMber?](#) - Returns the Trace Number of the selected trace.
- [CALC<cnum>:PAR:WNUMber?](#) - Returns the window number of the selected trace.
- [SYSTem:ACTive:CHANnel?](#) - Returns the number of the active channel. The active channel is the channel number that contains the active measurement.
- [SYSTem:ACTive:MEAS?](#) - Returns the name of the active measurement. As seen on the screen, the Active measurement has the highlighted Tr#.

- [SYSTem:CHANnels:CATalog?](#) - Returns the channel numbers currently in use.
 - [SYSTem:WINDows:CATalog?](#) - Returns the window numbers that are currently being used.
 - [SYSTem:MEAS:CATalog? \[chan\]](#) - Returns ALL measurement numbers, or optionally measurement numbers from a specified channel.
 - [SYSTem:MEAS<n>:NAME?](#) - Returns the name of the specified measurement (Tr#) number.
 - [SYSTem:MEAS<n>:TRACe?](#) - Returns the trace number of the specified measurement number.
 - [SYSTem:MEAS<n>:WINDow?](#) - Returns the window number of the specified measurement number.
-

Last modified:

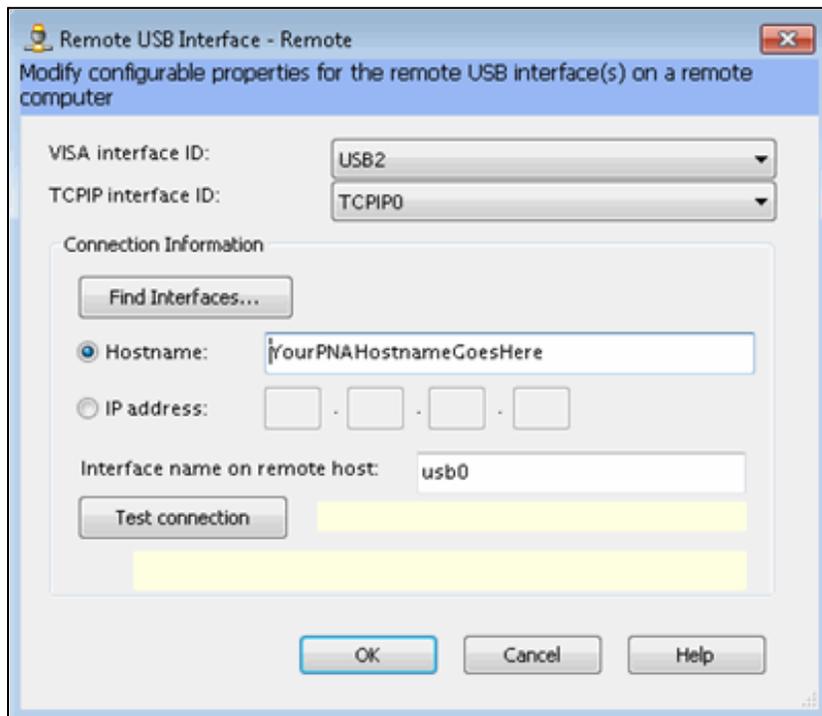
26-Feb-2013 Added Active Meas
7-Aug-2012 New topic

Remote Control of SCPI USB Devices Connected to a PNA

The following procedure is the USB equivalent of the PNA "[GPIB Pass-Through](#)" feature. This allows you to send SCPI commands from a remote PC to a USB device (such as a USB power sensor) that is connected to the PNA.

The PC must have the Agilent I/O Libraries installed.

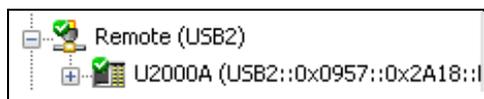
1. On the PNA, press **SYSTEM**, then [**Configure**], then [**SICL/GPIB**], then [SICL Enabled](#).
2. On your PC, start Agilent Connection Expert (ACE) which is the wizard for Agilent I/O Libraries.
3. In the ACE dialog, click **Add Interface**.
4. Click **Remote USB** then click **Add**.



5. In the Remote USB Interface dialog (shown above), do **EITHER** of the following:
 - Select **Hostname**, then enter the [Full Computer Name](#) of the PNA.
 - Select **IP address**, then enter the [IP address](#) of the PNA.

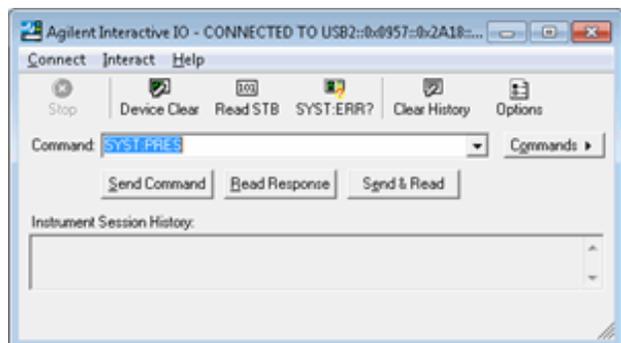
On the PC, the new USB interface in ACE should show the USBTMC-compliant devices (such as the U2000-Series sensors) that are connected to that PNA. In the above dialog, the VISA interface

ID is **USB2**. Therefore, a VISA program on the PC could send commands to the sensor that is connected to the PNA at the VISA resource string beginning with **USB2::<interface>**.



The above image shows a portion of the ACE dialog after adding the remote USB interface.

6. Right-click on the device you want to communicate with (U2000A in this example) then click **Send Commands To This Instrument**.



7. Type a command, then click **Send Command** or **Read Response**.

Last Modified:

14-Oct-2011 New topic

Configure for SCPI LAN using SICL / VISA

- [PNA Supported Interfaces](#)
- [Agilent I/O Libraries](#)
- [SICL / VISA Programs Running on the PNA](#)
- [Configure the PNA for SICL / VISA](#)
- [Configure the External Controller](#)

Other Topics about GPIB Concepts

PNA Supported Interfaces

The PNA supports the following interfaces for SICL / VISA communication:

- **LAN** - as a remote GPIB interface. The PNA LAN is presented as a virtual GPIB interface. It does NOT support simple TCPIP-based control. Therefore, when configuring the Agilent IO libraries on your PC, add a **REMOTE GPIB** interface, which uses the LAN client interface.
- **GPIB** - requires that your external controller have a GPIB card.

Note: For optimum LAN interface performance, use [COM](#) to control the PNA. SCPI commands can be sent to the PNA using the COM [SCPIStringParser](#) object.

The following interfaces are NOT supported:

- **USB**
- **Serial**

Important Note:

To enable VISA or SICL communication over LAN, you must do the following:

1. On the PNA, click **System**, point to **Configure**, then click **SICL/GPIB**.
2. Check **SICL Enabled**. To automatically enable SICL when the PNA is booted, check **Automatically enable on Startup**.
3. Click **OK**.

The PNA is now ready to be controlled over LAN.

[Learn more about this dialog box.](#)

Agilent I/O Libraries

The Agilent I/O libraries includes the drivers to allow you to communicate with Agilent test instruments. Every PNA is shipped with the Agilent I/O libraries installed. We recommend you do NOT upgrade the Agilent I/O libraries on the PNA as unexpected results may occur. If you choose to upgrade the Agilent I/O libraries on the PNA, do NOT change the default folder path in the InstallShield Wizard.

To communicate with the PNA, the Agilent I/O libraries must also be installed on your external controller. To purchase the Agilent I/O libraries, or download a free upgrade, go to www.agilent.com and search for IO Libraries. Scroll to find Software, Firmware & Drivers.

SICL / VISA Programs Running on the PNA

You can run your SICL / VISA program on the PNA to control the PNA. Although the Agilent I/O libraries are already installed on the PNA, it is configured as the **Host**. You must also configure a SICL or VISA LAN **Client** interface on the PNA, specifying the LAN hostname of that same PNA.

If your program uses the COM interface to VISA, and is compiled on a PC with the Agilent IO Libraries Suite (version 14 or later), and the resulting executable is copied and run on the PNA, it will produce a "type mismatch error". This is because the PNA has the 'M' version of Agilent I/O libraries. The following Visual Basic code is an example of how to avoid this error when communicating with the PNA from within the PNA:

```
Dim rm As IResourceManager
Dim fmio As IFormattedIO488
Set rm = CreateObject("AgilentRM.SRMClS")
Set fmio = CreateObject("VISA.BasicFormattedIO")
```

```
Set fmio.IO = rm.Open("GPIB0::22")
fmio.WriteString "*IDN?" & Chr(10)
MsgBox fmio.ReadString()
```

Controlling the PNA over LAN while controlling other instruments over GPIB

The PNA can NOT be both a controller and talker/listener on the same GPIB bus. Using SICL / VISA, you can use LAN to control the PNA, leaving the PNA free to use the rear-panel GPIB interface to control other GPIB devices.

Configure the PNA for SICL / VISA

1. On the PNA, click **System** then check **Windows Taskbar**
2. Click **Start** then point to **Program Files, Agilent IO Libraries**, then click **IO Config**
3. Select each GPIB Interface and click **Edit** to verify (or make) the default settings in the following table. These settings are REQUIRED when using a [82357A USB / GPIB](#) Interface with the PNA.
4. When complete, click **OK** to close the edit dialog.
5. Click **OK** to close the IO Config dialog.

VISA Interface Name	SICL Interface Name	Dialog box title	Description
GPIB0	gpib0	GPIB Using NI-488.2	PNA Rear-panel GPIB connector. This GPIB interface can be used to control the PNA OR for the PNA to control external equipment. IT CAN NOT DO BOTH IN THE SAME PROGRAM. Learn more about pass-through options.
GPIB1	hpib7	Internal Instrument Configuration	Internal interface for programs running on the PNA to control itself.
GPIB4	inst0	Internal Instrument Configuration	Used for LXI compliance . Do NOT delete this interface.

Configure the External Controller

Please refer to the Agilent I/O libraries documentation to learn how to configure your controller to communicate with the PNA. These links can show you how to find the following PNA information:

- [PNA full computer name](#)
- [GPIB Address](#)
- [IP Address](#)

This [example program](#) can help test your VISA configuration.

Last Modified:

13-Aug-2008 Added GPIB4

Agilent VEE Pro RunTime Installed

Beginning in Dec. 2005, Agilent VEE Pro RunTime is installed on new PNAs. This means that programs written with Agilent VEE (.vxe files) can be run directly on the PNA.

PNAs **without** Agilent VEE installed can go to the [Agilent VEE website](#) and download Agilent **VEE Pro 6.2** RunTime to the PNA and begin to run VEE programs directly on the PNA. This version does not require Agilent I/O Libraries suite 14. [Do **NOT** upgrade to Agilent I/O libraries suite 14 on the PNA.](#)

With Agilent VEE Pro RunTime installed on the PNA, the following examples can be run directly on the PNA:

- [Basic Control](#) of the PNA

For more VEE examples, see the [PNA support website](#).

For more information on Agilent VEE, see www.agilent.com/find/VEE

Basic Control using VEE

This VEE Pro 6.0 example does the following:

- Controls PNA windows and traces.
- Changes stimulus settings.
- Measures all four S parameters.
- Create markers and displays marker readout.

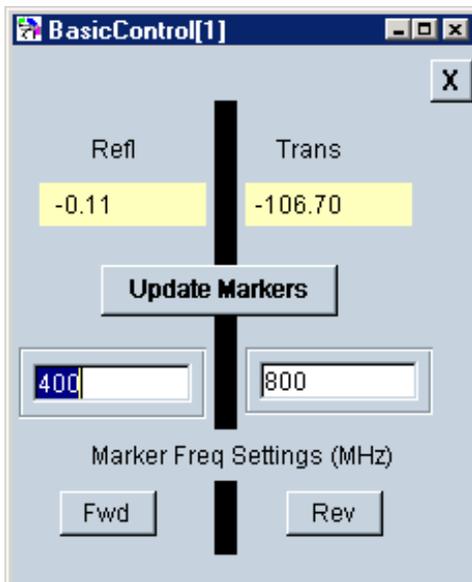
If this Help file is on a PNA and [VEE Pro RunTime is installed](#), then:

1. [Run the BasicControl.vxe example](#)
2. Then click **Open** on the following dialog box to run the program.

Otherwise, you can modify the example program using VEE, [save the VEE BasicControl.vee](#)

[Learn how to run this program as a Macro on the PNA.](#)

The following dialog box will be visible on the PNA when the example program is running.



- Click **Fwd** to activate the Forward (S11 and S21) measurements.
- Click **Rev** to activate the Reverse (S22 and S12) measurements.

- Click **Update Markers** to sweep the PNA.
- Type values to change Marker Frequencies.

ECal with Confidence Check using VEE

This VEE Pro 6.0 example performs an ECal and subsequent ECal confidence Check.

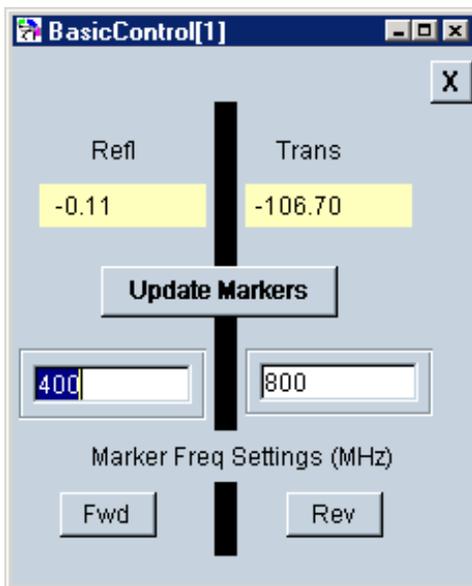
If this Help file is on a PNA and [VEE Pro RunTime is installed](#):

- [Run the .vxe example](#)
- Then click **Open** on the following dialog box to run the program.

Or to modify the example program using VEE, [save the VEE BasicControl.vxe](#)

[Learn how to run this program as a Macro on the PNA.](#)

The following dialog box will be visible on the PNA when the example program is running.



- Click **Fwd** to activate the Forward (S11 and S21) measurements.
- Click **Rev** to activate the Reverse (S22 and S12) measurements.
- Click **Update Markers** to sweep the PNA.
- Type values to change Marker Frequencies.

Interface Control

The Interface Control feature allows you to send remote commands and data to the following PNA rear-panel Interfaces: GPIB, Material Handler I/O, Test Set I/O, and Auxiliary I/O.

- [Overview](#)
- [How to Access Interface Control Settings](#)
- [Interface Control Dialog Box](#)
- [Z5623A H08 Test Set Commands](#)

Other System Configuration Topics

Note: Interface Control can be used ONLY with Standard channels; NOT with [Application](#) channels.

Overview

The Interface Control feature allows you to send data to control external equipment such as GPIB instruments, a material handler, test set, or other equipment, without needing to create a remote program. The PNA manages the timing and required interface setup. See [Rear Panel Tour](#).

- A unique set of control data can be sent for each channel. In addition, a unique set of control data can be sent before the channel sweep starts, and after the sweep ends.
 - Interface Control settings can be saved and recalled from the [Interface Control dialog box](#), or with [Instrument State Save and Recall](#).
 - Interface Control settings can be copied to other channels using [Copy Channels](#).
 - Control data can only be WRITTEN to the interfaces, NOT READ from the interfaces.
 - Control data is sent in the following order. This order cannot be changed.
1. [GPIB Interface](#)
 2. [Material Handler Interface](#)
 3. [Test Set Interface](#)
 4. [Aux Interface](#)
 5. [Dwell Time](#)

How to access Interface Control settings

Using front-panel HARDKEY [softkey] buttons

1. Press TRACE/CHAN
2. then [Channel]
3. then [Hardware Setup]
4. then [Interface Control]

Using a mouse with PNA Menus

1. Click Trace/Chan
2. then Channel
3. then More
4. then Interface Control

Programming Commands

Interface Control dialog box help

Interface Control

Enable Interface Control

Channel **1** Channel Control Label:

Before Sweep Start | After Sweep End

GPIB Commands - BEFORE Enable Control

Handler I/O Control Enable Control

Port A (0-255)

Port B (0-255)

Port C (0-15)

Port D (0-15)

Test Set I/O Control (addr.data) Enable Control

Dwell After Command ms

Aux I/O Output Voltage Enable Control

DAC1 (-10V -> +10V)

DAC2 (-10V -> +10V)

Reset All Save Recall **OK** Cancel Help

[See Interface Control Overview \(scroll up\)](#)

An [Instrument Preset](#) will reset all of the fields to their default settings.

Note: If an error is encountered when sending Interface Control data, an error message is displayed on the PNA screen. The [Channel Trigger State](#) is set to Hold. You must fix the condition that caused the error, then change the Channel Trigger State to its original setting.

Enable Interface Control Enables and disables ALL Interface Control communication. When cleared (default setting) Interface Control is disabled and NO data is sent. To send data, the individual interfaces must also be enabled.

Channel Specifies the channel number for dialog settings. Each channel is configured individually. The list box shows the channels that currently have measurements. There must be at least one measurement present in order to make settings.

Channel Label Specifies the label to be displayed on the second status bar at the bottom of the PNA screen. This field is shared with [External Testset control](#). The second status bar is automatically displayed when Interface Control is enabled.



[Learn about the primary status bar.](#)

Before Sweep Start - After Sweep End Tabs

Commands / data for all four interfaces can be sent both Before Sweep Start and After Sweep End. However, they are configured and enabled on separate tabs of the Interface Control dialog box. For example, to send GPIB commands both Before and After a PNA sweep, the Enable Control checkbox must be selected and commands entered on BOTH the Before Sweep Start and After Sweep End tabs.

Before Sweep Start The data is sent BEFORE the first trace on the channel begins sweeping.

After Sweep End The data is sent AFTER the last trace on the channel completes sweeping.

GPIB Commands

Notes:

- GPIB instruments CAN be connected to the PNA using a [USB/GPIB adapter](#).
- GPIB Queries are NOT supported.

Enable Control Enables and disables sending commands out the GPIB interface.

Multi-line edit control Each line contains a GPIB command using the following syntax:

```
address  command
```

Where:

address a number between 0 and 31. The PNA will look through all of the GPIB interfaces for an instrument connected to the specified address. If an instrument with that address is not recognized, an error is returned.

command a GPIB command, with or without enclosing quotes. Enclosing quotes are ignored.

Address and command are separated by at least one space.

Commands should be separated by a new line, or carriage return. For example:

```
19 ":init:cont off"  
16 init:imm
```

The front-panel **Enter** key inserts a new line into the field.

The number of GPIB commands that can be entered is limited only by the available memory of the PNA.

See [Z5326A H08 Test Set Commands](#).

Material Handler I/O

Enable Control Enables and disables sending data out the [Material Handler I/O connector](#)

Ports A, B, C, D Sends values to the respective Handler I/O port. Although ports C and D are normally bidirectional, ONLY Output mode is allowed using the Interface Control feature. It cannot read from these, or any other, ports.

Test Set I/O

Note: The PNA has a separate interface for [controlling the E5091A Test Set](#).

Enable Control Enables and disables sending data out the [External Test Set I/O connector](#).

Multi-line edit control Each line contains a Write command using the following syntax:

```
address.value
```

Where:

address any positive integer.

value numeric character. Entries that require **alpha** characters should use the [GPIB](#)

[interface](#).

Address and value are separated by a period. For example:

```
18.2  
27.3
```

Entries should be separated by a new line, or carriage return. The PNA front-panel **Enter** key inserts a new line into the field.

All entries are sent out the Test Set I/O port using the [WriteData Method](#).

The number of entries is limited only by the available memory of the PNA.

Aux I/O (NO longer available)

Enable Control Enables and disables sending data out the Auxiliary I/O connector.

DAC1, DAC2 Sets voltages on the Aux I/O connector pins 2 (DAC1) and pin 3 (DAC2).

Dwell After Command Specifies a wait time, in milliseconds, after all commands to all interfaces are sent. Any positive integer is allowed. This is used to allow all external devices to settle before beginning a measurement. An erratic trace could indicate that more settling time is necessary.

Reset All Sets ALL fields on ALL channels to their default values.

Save and Recall Saves and recalls the contents of this dialog box. If the Interface Control dialog box is populated with settings during an [Instrument State Save](#), the settings are automatically recalled with the Instrument State settings.

Interface control uses an *.xml file type. An example file is stored on the PNA hard drive. You can recall it into the dialog, or you can open and edit it with a word processor, such as Word Pad.

OK Applies the settings and closes the dialog box.

Cancel Does not apply changes that were made, and closes the dialog box.

Z5623A H08 Test Set Commands

The following table lists the commands that are used to control the popular Agilent Z5623A H08 Test Set. These commands can be entered into the [GPIB Interface](#) control.

Connection Path	Test Set Command
Reflection to Port 1	refl_01
Reflection to Port 2	refl_02
Reflection to Port 3	refl_03
Reflection to Port 4	refl_04
Reflection to Port 5	refl_05
Reflection to Port 6	refl_06
Reflection to Port 7	refl_07
Reflection to Port 8	refl_08
Transmission to Port 1	tran_01
Transmission to Port 2	tran_02
Transmission to Port 3	tran_03
Transmission to Port 4	tran_04
Transmission to Port 5	tran_05
Transmission to Port 6	tran_06
Transmission to Port 7	tran_07
Transmission to Port 8	tran_08
Reset	*rst
Reflection Termination	*r_term
Transmission Termination	*t_term
All Termination	*all_term

Last Modified:

14-Jul-2013 Removed old content (BH) and added std channels
4-Sep-2008 Removed legacy content

External Test Set I/O Connector

General Description

This DB-25 female connector is used to control external test sets. The external test set bus consists of 13 multiplexed address and data lines, three control lines, and an open-collector interrupt line. The Test Set IO is not compatible with the 8753 test sets.

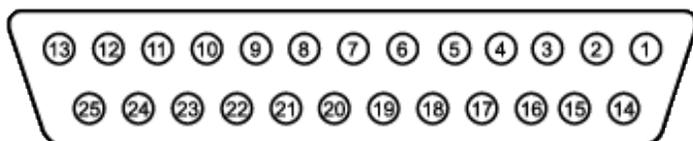
You can change the settings on the External Test Set IO connector through [SCPI](#) and [COM](#) programming commands. The settings are NOT accessible through the front-panel keys or display menu.

Notes:

- The External Test Set pin settings are NOT affected by [Instrument State Save/Recall](#) or [Instrument Preset](#).
- At PNA Power Up and return from Hibernation, the External Test Set bus data lines, address lines, and control lines are set HIGH, and no strobe lines are pulsed.

Caution: Do not mistake this connector with a Parallel Printer port. A printer may be damaged if connected to this port.

Other System Configuration Topics



Pin	Name	Description
1	SEL0	Test set select bit 0; tied to GND
2	Sweep Holdoff In	TTL input - state may be read with SCPI or COM command
3	AD12	Address and latched data
4	AD10	Address and latched data
5	AD9	Address and latched data
6	AD8	Address and latched data

7	GND	0V
8	LAS	TTL output Low = Address Strobe
9	AD4	Address and latched data
10	AD3	Address and latched data
11	AD2	Address and latched data
12	GND	0V
13	Interrupt In	TTL input - state may be read with a SCPI or COM command
14	No connect	CAUTION: Older PNAs have +22v on this line; this will damage a printer.
15	SEL1	Test set select bit 1; tied to GND
16	SEL2	Test set select bit 2; tied to GND
17	AD11	Address and latched data
18	SEL3	Test set select bit 3; tied to GND
19	AD7	Address and latched data
20	AD6	Address and latched data
21	AD5	Address and latched data
22	AD0	Address and latched data
23	AD1	Address and latched data
24	LDS	TTL output - active low data strobe
25	RLW	TTL output - high-read, low write

SEL0-SEL3 (pins 1,15,16,18)

Description

Selects addresses of test sets that are "daisy chained" to this port. The select code is set to zero at the PNA connector and is incremented by one as it goes through each successive external test set. Therefore, the first test set in the chain has address zero and so on, for up to 16 test sets.

HW Details

Connected to ground inside the PNA.

Timing

None

Sweep Holdoff In (pin 2)

Description

Input line used by the test set for holding off a sweep. Holding off a sweep is one way of introducing a delay that allows an external device to settle before the PNA starts taking data. You must write a program that will query the line and perform the delay. The program needs to query the line and keep PNA from sweeping while the line remains low. When a subsequent query detects that the line went high the program would then trigger the PNA to start the sweep.

Use either Single or External trigger mode to control the PNA sweep.

HW Details

This pin has a series 215-ohms resistor followed by 4.7k-ohm pull-up and then an "ABT" TTL buffered register.

Timing

This input is not latched by the PNA hardware. Therefore the input level must be held at the desired state by the test set until it's read by your program.

AD0-AD12 (pins 3-6, 9-11, 17, 19-23)

Description

Thirteen lines are used to output data addresses or input / output data. Several [SCPI](#) and [COM](#) commands are available for reading and writing to these lines. You can choose to use commands where the PNA provides the appropriate timing signals needed for strobing the addresses and data. Or you can choose to control the timing signal directly. The timing signals are RLW, LAS and LDS. If you decide to do direct control refer to the corresponding SCPI and COM command details. Close attention to detail is needed to insure the desired results.

After a write command, lines AD0-AD12 are left in the state they were programmed. Default setting for Mode is Read / Input).

After a read command, lines AD0-AD12 are left in input mode. While in this mode an external test set attached to the IO is free to set the level on each line.

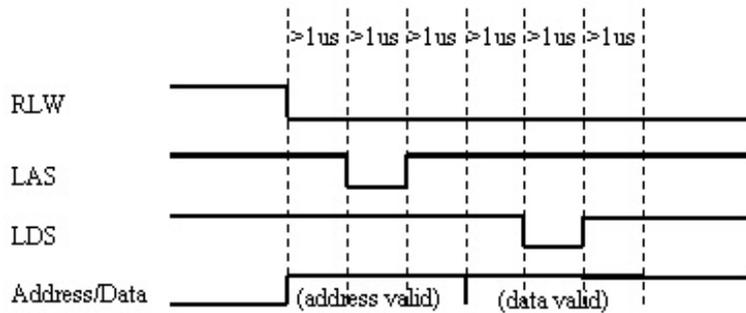
HW Details

Each of these I/O pins has a series 215-ohm resistor followed by 4.7k-ohm pull-up resistor.

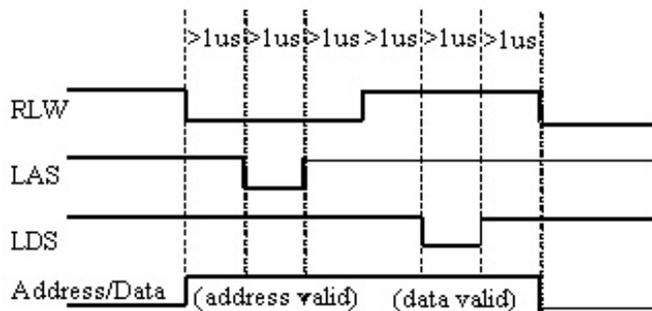
Write/Read is implemented by an output tri-state TTL buffer / latch for latching and enabling write data in parallel with a TTL input buffer for reading.

Timing

Output Address and data setup and hold times are 1us minimum.



Address & Data I/O Write



Address & Data I/O Read - Data must be valid for 1us before and after strobe

GND (pins 7, 12)

Description

Two ground pins used as ground references by the test set.

HW Details

Connected to digital ground.

Timing

None.

LAS (Low Address Strobe) (pin 8)

Description

This line has two behaviors that are command dependent. Refer to the [SCPI](#) and [COM](#) commands for further details.

In one behavior LAS is one of the lines used by the PNA to provide appropriate timing for writing Address and Data to the

Test Set. In this case LAS is controlled automatically by the PNA and is intended to be used as the strobe for the Address. When LAS is low, lines AD0 - AD12 represent the Address. LAS will return to its normally high state when the transaction is finished.

In the second behavior the PNA will NOT provide appropriate timing. In this case LAS is controlled directly by the user through a [SCPI](#) or [COM](#) command. When the transaction is finished LAS is left set to the state it was programmed to until another command changes it. (Default for LAS is TTL High).

HW Details

This output pin is driven by a TTL latched buffer with a series 215-ohm resistor followed by 2.15k-ohm pull-up.

Timing

Strobe length, setup and hold times are all 1us minimum.

See the description for [AD0-AD12](#) for more timing information.

Interrupt In (pin 13)

Description

Query this line with a [SCPI](#) or [COM](#) command.

HW Details

This line is a non-latched TTL input, has series 215-ohms followed by 4.64k-ohm pullup.

Timing

The Test Set must maintain at the desired TTL level until its read.

(pin 14) No Connect (previously +22V)

WARNING: Early versions of the PNA had +22v on this pin. **Connecting a printer to this port will usually damage the printer.**

Description

+22V, 100mA max. The 25-pin D connector is the same as a computer parallel printer port connector. Pin (14) corresponds to a printer's "autofeed" line. **Connecting a printer to this port will damage the printer if +22v is present** since printers requires less than 5V on all control lines.

HW Details

No connect

Timing

None

LDS (Low Data Strobe) (pin 24)

Description

This line has two behaviors that are command dependent. Refer to the External Test Set IO SCPI and COM commands for further details. (Default setting for LDS is TTL High)

In one behavior LDS is one of lines used by the PNA to provide appropriate timing for writing Address and Data to the Test Set. In this case LDS is controlled automatically by the PNA and is intended to be used as the strobe for the Data. When LDS is low, lines AD0 - AD12 represents Data. LDS will return to its normally high state when the transaction is finished.

In the second behavior the PNA will NOT provide appropriate timing. In this case LDS is controlled directly by the user through a SCPI or COM command. When the transaction is finished the LDS is left set to the state it was programmed to.

HW Details

This output pin is driven by a TTL latched buffer with a series 215-ohm resistor followed by 2.15k-ohm pull-up.

Timing

Strobe length, setup and hold times are all 1us minimum.

See the description for [AD0-AD12](#) for more timing information.

RLW (pin 25)

Description

This line is the output for the Read Write signal. It has two behaviors that are command dependent. Refer to the External Test Set IO SCPI and COM commands for further details. (Default setting for RLW is TTL High)

In one behavior RLW is controlled automatically by the PNA during a Read Write operation. When RLW is low, lines AD0 - AD12 represent output Data. When RLW is high, the lines represent input Data.

In the second behavior the PNA does NOT provide the timing. The user must control it directly through the SCPI or COM command. In this case the line is left set to the state it was programmed to.

HW Details

This pin is a TTL latched output with a series 215-ohm resistor followed by 2.15k-ohm pull-up resistor.

Timing

Strobe length, setup and hold times are all 1us minimum.

See the description for [AD0-AD12](#) for more timing information.

Material Handler I/O Connector

This [rectangular 36-pin female connector](#) provides communication signals between the PNA and a material parts handler. You can change the settings on the Material Handler IO connector using [SCPI](#) and [COM](#) commands. The settings are NOT accessible through the front-panel keys or display menu.

- [Overview - Controlling a Material Handler](#)
- [Pin Assignments](#)
- [Pin Descriptions](#)
- [Timing Diagrams](#)
- [Input Output Electrical Characteristics](#)

Note: On early PNAs this connector is labeled "GPIO". It is covered to indicate that the connector is not functional.

Overview - Controlling a Material Handler

The PNA is capable of interacting with an external material handler or part handler. This allows the PNA to be used in an automated test environment, where devices to be tested are inserted into a test fixture by a part handler, and sorted into pass/fail bins by the handler after testing is complete. By connecting the part handler to the PNA Auxiliary or Material Handler I/O ports, the PNA and part handler can synchronize their activities in a way that makes automated testing possible.

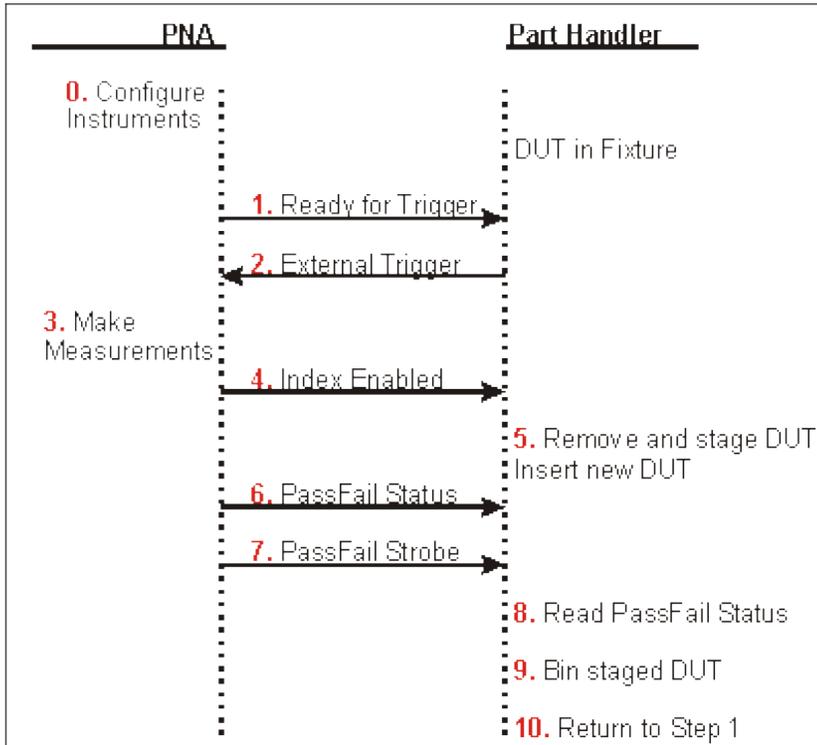
PNA and Part Handler Preparation

1. [Define the measurements](#) you want to make.
2. [Define limits](#) for each of the measurements.
3. Configure the PNAs Material Handler port so that it is compatible with your part handler. This usually involves setting the [handler logic](#), [pass/fail logic](#), [pass/fail scope](#), and [pass/fail mode](#). These settings are made remotely using [SCPI](#) or [COM](#) commands.
4. Use a cable to connect the PNA to your part handler.
5. Put the PNA in [External Trigger](#) mode.
6. Load parts in handler per manufacturer instructions.

Note: The Material Handler configuration settings REMAIN after an [Instrument Preset](#). The settings will revert to their default settings ONLY after the PNA is restarted, or until they are changed by you. Material Handler settings are [saved and recalled with Instrument State](#).

Flow Diagram

The following diagram and descriptions summarizes the events that occur during automated testing. 'DUT' refers to Device Under Test.



Text Descriptions

0. (Optional). The PNA sends values out the Material Handler to configure external instruments. The A,B,C, and D ports of the Material Handler can be used to control devices used in testing, such as step attenuators, part handlers, or even the DUT itself. Also, the DAC1 and DAC2 lines on the [Power I/O connector](#) can be used to provide bias voltages for devices and instruments. If you wish to use the Material Handler for testing, you will need to write a program to send values out the various lines and ports, as there is no activity on these lines by default.
1. The part handler receives a [Ready for Trigger](#) signal from the PNA. This indicates that the PNA is properly configured and ready to take a measurement.
2. The part handler sends an [External Trigger](#) signal to the PNA. This signals that the part handler has settled, and allows the PNA to begin taking measurements.
3. The PNA takes measurements on all triggerable channels.
4. The [Index line](#) on the material handler goes to a Low state, which means that all required data has been collected by the PNA.
5. The part handler removes the DUT from the test fixture, and inserts a new DUT into the fixture. This operation is often referred to as part handler indexing. The device just tested is staged (removed from the

fixture and prepared for binning), and the next part to be tested is put into the fixture. The removed DUT cannot be assigned to a Pass/Fail bin yet, as the Pass/Fail status is not available.

6. The PNA sends the [Pass/Fail Status](#).
7. The PNA sends the [Pass/Fail Strobe](#) meaning that the Pass/Fail status has been determined.
8. The part handler reads the Pass/Fail Status line.
9. The part handler bins the staged part based on the Pass/Fail Status.
10. The test process repeats at step 1, waiting for Ready for Trigger from the PNA.

Material Handler IO Pin Assignments



Pin	Description
1	Ground
2	/ INPUT1
3	/ OUTPUT1
4	/ OUTPUT2
5	/ Output port A0
6	/ Output port A1
7	/ Output port A2
8	/ Output port A3
9	/ Output port A4
10	/ Output port A5
11	/ Output port A6
12	/ Output port A7
13	/ Output port B0
14	/ Output port B1
15	/ Output port B2

16	/ Output port B3
17	/ Output port B4
18	/ Ext. Trigger
19	/ Output port B5
20	/ Output port B6 -or / Index Signal Learn more
21	/ Output port B7 or / Ready for Trigger Learn more
22	/ In/Out port C0
23	/ In/Out port C1
24	/ In/Out port C2
25	/ In/Out port C3
26	/ In/Out port D0
27	/ In/Out port D1
28	/ In/Out port D2
29	/ In/Out port D3
30	Port C Status
31	Port D Status
32	/ Output Port Write Strobe
33	/ Pass/Fail
34	/ Sweep End
35	+5V
36	/ Pass/Fail Write Strobe

Pin Descriptions

Input1

When this Input line receives a Low pulse from the material handler, data is latched on the [OUTPUT1 and OUTPUT2](#) lines. See [OUTPUT1|2 Data Output Write Timing](#)

The Input Line activity can be read:

	SCPI	COM
	CONTrol:HANDler:INPut?	get_Input1 Method

Output1, Output2

See [OUTPUT1|2 Data Output Write Timing](#)

The **current** state of these latched TTL outputs may be set High or Low (Default setting) using the (non-user) [SCPI put_Output \(COM\)](#) commands.

The **next** state (following a negative edge on the INPUT1 line) may be pre-loaded to High or Low (Default setting) using the user commands.

For example, on the next negative pulse on the INPUT1 line, you want the OUTPUT1 line to go from 0 to 1. To do this:

```
CONT:HAND:OUTP1:DATA 0 'Force the OUTPUT1 line to 0  
CONT:HAND:OUTP1:USER 1 'Set the OUTPUT1:USER buffer to 1, indicating the next  
state
```

	SCPI	COM
Write User Data	CONT:HAND:OUTP<pin>:USER	put_Output Method
Read last value written	CONT:HAND:OUTP<pin>:USER	get_Output Method.
Write non-user data	CONT:HAND:OUTP<pin>:DATA	put_Output Method
Read last value written	CONT:HAND:OUTP<pin>:DATA	get_Output Method

Output Ports A and B

These two general purpose, 8-bit output ports are used to write data to the material handler. When any line changes state, all output lines are latched to the I/O connector as the [Output Write Strobe](#) goes Low.

The default state for data is Low.

[See Data Output Write Timing Diagram](#)

Set Port Logic:

The logic for the data lines can be set to either: Positive (1 = High) or Negative (1 = Low). This setting affects all data ports. They cannot be set independently.

SCPI
[CONTrol:HANDler:LOGic](#)

COM
[PortLogic Property](#)

Combine to read or write data to Port F:

Ports A and B can be virtually combined to write data to one 16-bit I/O port F.

SCPI
[CONTrol:HANDler:F <num>](#)

COM
[put Port \(F\)](#)

Input/Output Ports C and D

These two general purpose 4-bit Input/Output ports are used to write data (Output) or read data (Input). These lines could be used to write to an external device such as a step attenuator.

When any line changes state, all output lines are latched to the I/O connector as the [Output Write Strobe](#) goes Low. [See Data Output Write Timing](#).

Set Input | Output Mode:

Each port may be independently defined as Output or Input.

SCPI
[CONTrol:HANDler:C:MODE](#)
[CONTrol:HANDler:D:MODE](#)

COM
[PortMode Property](#)

Set Port Logic:

The logic for the data lines can be set to either: Positive (1 = High) or Negative (1 = Low). This setting affects all data ports. They cannot be set independently.

SCPI
[CONTrol:HANDler:LOGic](#)

COM
[PortLogic Property](#)

Read or write data:

Ports C and D can be virtually combined to read or write data to one 8-bit I/O port E. When combined, **both** C and D ports must be set to either INPUT or OUTPUT mode.

SCPI
[CONTrol:HANDler:<port>\[:DATA\]>](#)

COM
[get_Port\(x\)](#)
[put Port \(x\)](#)

Port C Status, Port D Status

These two output lines indicate the Read / Write mode of the C and D ports.

- A Low level indicates that the associated port is in **INPUT** mode (read only).
- A High level indicates that the associated port is in **OUTPUT** mode (write only).

These logic of these status outputs cannot be changed.

See [Input/Output Ports C and D](#) to learn how to set I/O Mode

[See Data Output Write Timing](#)

Output Port Write Strobe

This Output line goes Low to write data from [Ports A and B](#) and [Ports C and D](#) when a change is detected on any of the data lines.

These logic of this strobe output cannot be changed.

[See Data Output Write Timing](#)

External Trigger

When trigger source is set to external, this Input line accepts a trigger signal from the material handler. This usually means that a part is in place and ready to be tested.

[See Trigger Timing Diagram](#)

Index

A Low signal on this Output line indicates to the material handler that the measurement is complete. This usually means that the handler can connect the next device. However, measurement data is not available until data is calculated. [see Trigger Timing Diagram](#).

Set Function:

This line also serves as a data line. Set the function using the following commands:

SCPI
[CONTrol:HANDler:INDex:STATe](#)

COM
[IndexState](#)

Ready for Trigger

When this output line goes low, it indicates to the material handler that the PNA is ready for a trigger signal.

[See Trigger Timing Diagram](#)

[See Pass/Fail Timing Diagram](#)

Set Function:

This line also serves as a data line. Set the function using the following commands:

SCPI
[CONTRol:HANDler:RTRigger:STATE](#)

COM
[ReadyForTriggerState](#)

Pass/Fail State

This Output line indicates to the handler whether the limit test has passed or failed.

Pass/Fail state is valid only when the [limit test](#) function is ON and while [Pass/Fail strobe](#) line is Low. [See Pass/Fail Timing Diagram](#)

Set Pass / Fail Logic:

- Positive Logic: High=Pass, Low=Fail. (Default setting)
- Negative Logic: High=Fail, Low=Pass.

SCPI
[CONTRol:HANDler:PASSfail:LOGic](#)

COM
[PassFailLogic Property](#)

Set Default Conditions:

- **PASS**- the line stays in PASS state. When a device fails, then the line goes to fail after the Sweep End line is asserted.
- **FAIL**- the line stays in FAIL state. When a device passes, then the line goes to PASS state after the Sweep End line is asserted.
- **No Wait**- the line stays in PASS state. When a device fails, then the line goes to fail IMMEDIATELY. (Default setting)

SCPI

COM

[CONTrol:HANDler:PASSfail:MODE](#)

[PassFailMode Property](#)

Set Pass / Fail Scope:

- **Channel scope:** The line resets to the default state after the measurements on a channel have completed.
- **Global scope:** The line resets to the default state after the measurements on all triggerable channels have completed. (Default setting)

SCPI

COM

[CONTrol:HANDler:PASSfail:SCOPE](#)

[PassFailScope Property](#)

Pass/Fail Write Strobe

A Low pulse indicates that [Pass/Fail](#) line is valid and the Pass / Fail State is output to the material handler.

The Pass/Fail Strobe is fixed in duration and timing. However, when the strobe occurs depends on the Pass/Fail Mode and Pass/Fail Scope (Channel or Global) settings. [See Pass/Fail State](#)

[See Pass/Fail Timing Diagram](#)

+5V

+5V nominal output (100mA max).

Protected by self-healing fuse.

Sweep End

This output line indicates the status of the PNA sweep. The sweep includes sweeping the source and taking data.

- **Low** (falling edge) indicates that the specified sweep event has finished. This does NOT indicate that all calculations have finished.
- **High** indicates that the specified sweep event is active.

[See Trigger Timing Diagram](#)

Set Sweep Event Mode:

- **Sweep:** indicates that a single source sweep has finished. (Default setting)
- **Channel:** indicates that a single channel has finished.
- **Global:** indicates that all enabled channels have finished.

SCPI

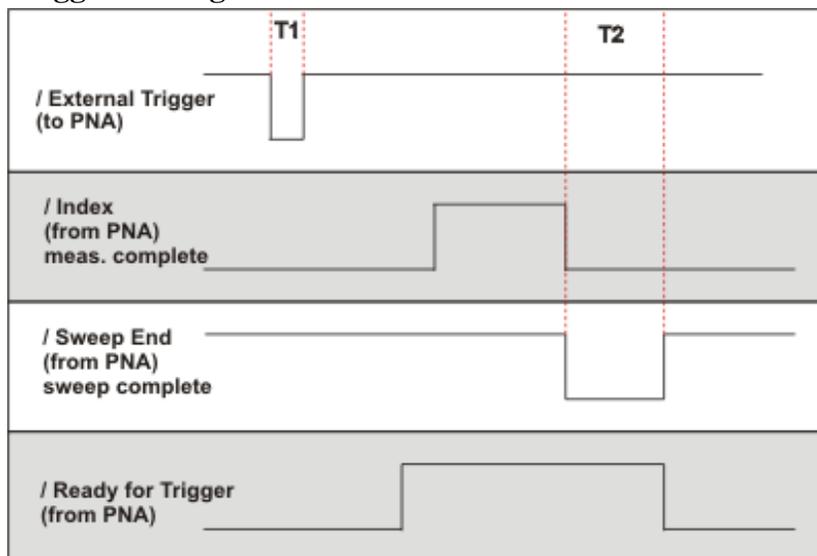
[CONTrol:HANDler:SWEepend](#)

COM

[SweepEndMode Property](#)

Timing Diagrams

Trigger Timing

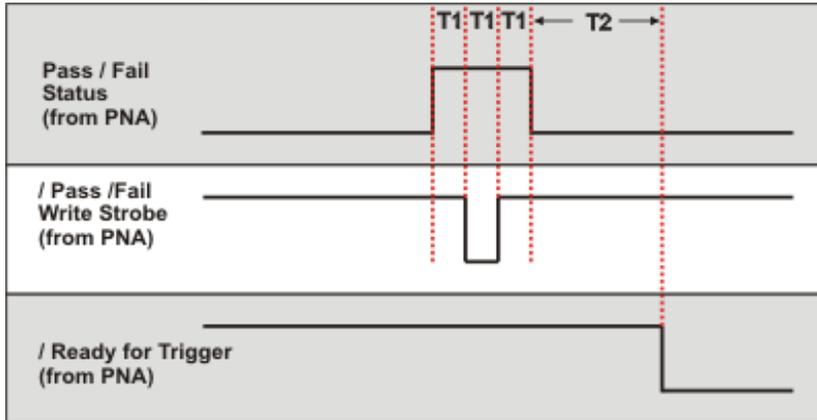


All signals are active low.

$T1 = 1 \mu s$ External Trigger pulse width

$T2 > 10 \mu s$ Sweep End pulse width (both High and Low)

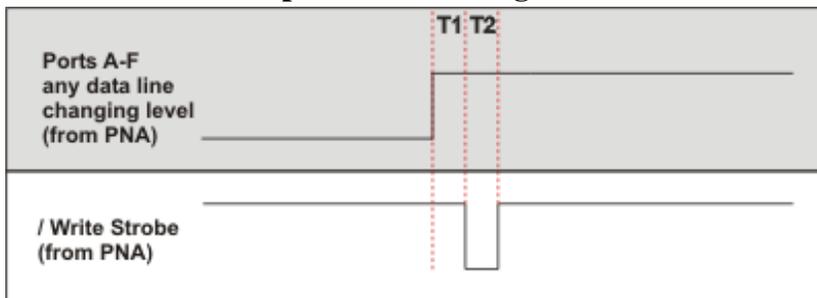
Pass / Fail Timing



$T1 = 1 \mu s$ Pulse width and response time of Pass / Fail Strobe

$T2 > 10 \mu s$ Ready for Trigger lag

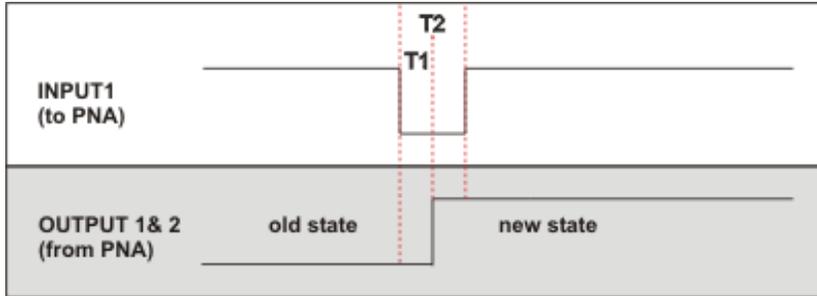
Ports A-F Data Output Write Timing



$T1 = 1 \mu s$ Write Strobe response time

$T2 = 1 \mu s$ Write Strobe pulse width

OUTPUT1|2 Data Output Write Timing



The old state to new state transition can be either low to high (as shown) or high to low.

$T1 = .6 \mu s$ [Output1|2](#) response time

$T2 = 1 \mu s$ [Input1](#) Strobe pulse width

Input / Output Electrical Characteristics

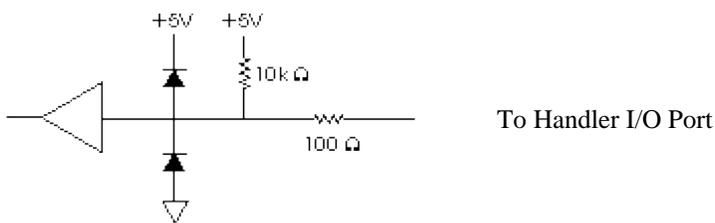
All Material Handler I/O Input and Output lines are TTL compatible.

Input and Input/Output lines

Lines carrying information IN (or bidirectional) to the PNA from the material handler.

Maximum Input Voltages:	-0.5 V to 5.5 V
TTL High level:	2.0 V to 5.0 V
TTL Low level:	0 V to 0.5 V

PNA Input and Input/Output Circuit Diagram



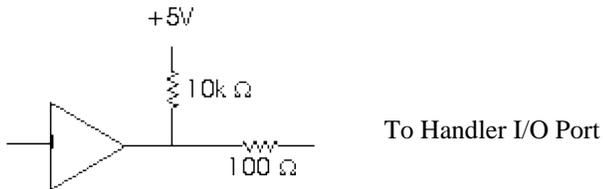
Note: The [INPUT1](#) line does NOT have the 10K pullup resistor.

Output Lines

Lines carrying information OUT of the PNA to the material handler.

Maximum Output Current:		-10 mA to 10 mA
Output Current	TTL High level:	-5 mA
	TTL Low level:	3 mA
Output Voltage	TTL High level:	2.0 V to 3.3 V
	TTL Low level:	0 V to 0.8 V

PNA Output Circuit Diagram



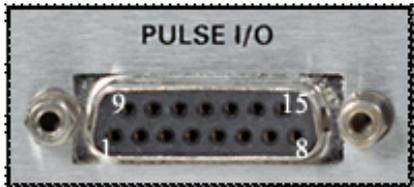
Last modified:

- 3-May-2012 Removed old types and hibernate
- 9-Apr-2012 Modified for all new models
- 6-Apr-2009 Replaced N5242A with PNA-X
- 8-Feb-2008 Clarified Types
- Nov 3, 2006 Added link from pin 20

Pulse I/O Connector

The models offer the 15 pin D connector provides access to Pulse Modulators and Generators.

- [See IF Path Configuration Dialog and block diagram](#), which includes the Pulse Modulators and Generators.
- [See the Integrated Pulsed Application](#)



Pin	Name	Description
1	IFGateAIn	IF pulse gate input A (TTL)
2	IFGateBIn	IF pulse gate input B (TTL)
3	IFGateCIn	IF pulse gate input C (TTL)
4	IFGateDIn	IF pulse gate input D (TTL)
5	IFGateRIn	IF pulse gate input R (TTL)
6	DCOM	Ground
7	PulseSyncIn	Pulse gen. synchronization trigger input (TTL)
8	RFPulseModIn	RF source pulse modulation drive input (TTL)
9	DCOM	Ground
10	Pulse1Out	Hardwired pulse train output #1 (TTL)
11	Pulse2Out	Hardwired pulse train output #2 (TTL)
12	Pulse3Out	Hardwired pulse train output #3 (TTL)
13	Pulse4Out	Hardwired pulse train output #4 (TTL)
14	N.C.	No connect -- for future use
15	DCOM	Ground

See Pulse [SCPI](#) and [COM](#) commands

N1966A Pulse I/O Adapter



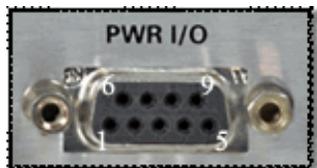
This D connector to RF adapter makes accessing the Pulse I/O connector more convenient.

Last Modified:

4-May-2011	Added N522x
16-Jul-2007	Clarification
18-Jan-2007	MX New topic

Power I/O Connector

The 9-pin D connector replaces much of the functionality of the AUX I/O connector on older PNA models. [See Rear Panel](#)



Pin	Name	Description
1	+15V	+15V @ 400 mA
2	-15V	-15V @ 400 mA
3	AnalogOut1	<p>Controlled from the DC Source Control dialog.</p> <p>Analog Output Voltage</p> <p>Programmable +/-10V @100 mA out</p> <p>Nominally 0 ohms</p> <p>2.44mV typical resolution</p> <p>1MHz BW</p> <p>Read and write voltage programmatically using:</p> <ul style="list-style-type: none"> • CONT:AUX:OUTP1:VOLT (SCPI - read and write) • get OutputVoltage Method (COM - read) • put OutputVoltage Method (COM - write)
4	AnalogOut2	<p>Controlled from the DC Source Control dialog.</p> <p>Analog Output Voltage</p> <p>Programmable +/-10V @100 mA out</p> <p>Nominally 0 ohms</p> <p>2.44mV typical resolution</p> <p>1MHz BW</p> <p>Read and write voltage programmatically using:</p>

		<ul style="list-style-type: none"> • CONT:AUX:OUTP2:VOLT (SCPI - read and write) • get OutputVoltage Method (COM - read) • put OutputVoltage Method (COM - write)
5	ACOM	System ground
6	GndSense	<p>Ground sense for Analog In & Out</p> <p>Connected with 51.1-ohm to ACOM</p> <p>Read voltage programmatically using:</p> <ul style="list-style-type: none"> • InputVoltageEX Property (COM) • CONT:AUX:OUTP3:VOLT (SCPI)
7	AnalogIn1	<p>Analog input:</p> <p>+/-10V @ 1.22mV typical resolution</p> <p>Rin >1 M-ohm</p> <p>BW ~ 1 MHz</p> <p>ADC conversion time < 1 us typical</p> <p>Read voltage programmatically using:</p> <ul style="list-style-type: none"> • CONT:AUX:INP1:VOLT (SCPI) • InputVoltageEX Property (COM) <p>Voltages on this pin can also be read using ADC receiver measurements. Learn more.</p>
8	AnalogIn2	<p>Analog input:</p> <p>+/-10V @ 1.22mV typical resolution</p> <p>Rin >1 M-ohm</p> <p>BW ~ 1 MHz</p> <p>ADC conversion time < 1 us typical</p> <p>Read voltage programmatically using:</p>

		<ul style="list-style-type: none"> • CONT:AUX:INP2:VOLT (SCPI) • InputVoltageEX Property (COM) <p>Voltages on this pin can also be read using ADC receiver measurements. Learn more.</p>
9	Power Button	<p>Open collector input</p> <p>Active low replicates power button key press.</p>

Last Modified:

9-Apr-2012 Removed AuxIO link
21-Feb-2012 Added source control dialog
4-May-2011 Modified for N522x
23-Mar-2009 Added links to ADC meas.
13-Aug-2008 Edited for InputVoltageEx
10-Jul-2007 Added COM commands
18-Jan-2007 MX New topic

New Programming Commands

The following are new programming commands for **PNA release A.10.00** See What's New .

Toolbars ON OFF	SCPI	COM
Show virtual hardkeys	DISPlay:TOOLbar:KEYS[:STATe]	app.ShowToolbar
Show port extensions	DISPlay:TOOLbar:EXT[:STATe]	app.ShowToolbar
Shown Time Domain Transform	DISPlay:TOOLbar:TRAN[:STATe]	app.ShowToolbar

Miscellaneous	SCPI	COM
Set and read marker bucket.	CALC:MARK:BUCKeT	Bucket Number
Read the directory path	SYST:CONF:DIR	DirectoryPath
Read supported parameters for a Meas Class	SYST:MCL:PAR:CAT?	None
System clock state	SYST:CLOCK	None
IOConfiguration	None	IOConfiguration Object
Read model number	None	ModelNumber
Read serial number	None	SerialNumber

The following are new programming commands for **PNA release A.09.90** See What's New .

SMC Phase Reference Cal	SCPI	COM
Unknown Mixer calibration state	SYST:CAL:PHAS:UNKN:INCLude	IncludeUnknownMixer
Unknown mixer LO frequency	SYST:CAL:PHAS:UNKN:LO:FREQUency	UnknownMixerLOFrequency
Unknown mixer LO power	SYST:CAL:PHAS:UNKN:LO:POWEr	UnknownMixerLOPower
Input power level to the unknown mixer	SYST:CAL:PHAS:UNKN:INPut:POWEr	UnknownMixerInputPower
Connector type and gender of your Cal Kit.	SYST:CAL:PHAS:CONNector	ConnectorType PhaseRef
Cal Kit to be used to perform the S-parameter Cal	SYST:CAL:PHAS:CKIT	CalKitType PhaseRef
Set the ports to be calibrated.	SYST:CAL:PHAS:PORT[X]	IncludePort
Reverse Port2 Coupler	SYST:CAL:PHAS:DEEMbed	DeembedCoupler

CalPod as ECal	SCPI	COM
Returns whether the specified device is a CalPod module	SENS:CORR:CKIT:ECAL:CHAR:INSitu:ENABLE?	SupportsInSituCharacterization
Sets or returns whether the specified device will be characterized as an in situ device.	SENS:CORR:CKIT:ECAL:CHAR:INSitu:STATE	InSituCharacterization

Read Date and Time	SCPI	COM
Read the last modified date of a Cal Set	CSET DATE?	None
Read the last modified time of a Cal Set	CSET:TIME?	None
Read the last modified date of a file	MMEM:DATE?	None
Read the last modified time of a file	MMEM:TIME?	None

Miscellaneous	SCPI	COM
Returns whether the specified external device responds.	SYSTEM:CONFIGure:EDEVICE:EXISTs?	

The following are new programming commands for **PNA release A.09.85** See What's New .

Get/Save Noise Parameters SNP Data	SCPI	COM
Read noise parameter SNP data	SENS:NOISe:SNP?	GetSnPData
Write noise parameters to SNP file	SENS:NOISe:SNP:SAVE	WriteSnPData

Modify TRL Cal Kit	SCPI	COM
Set reference plane	SENS:CORR:COLL:CKIT:TRL:RPLane	None
Set impedance standard	SENS:CORR:COLL:CKIT:TRL:IMPedance	None
Set LRL auto-characterization	SENS:CORR:COLL:CKIT:TRL:LRLChar	None

The following are new programming commands for **PNA release A.09.80** See What's New .

Calibrate All Channels

Select the channels to be calibrated.	SYST:CAL:ALL:CHAN:SEL	Channels
Set the IFBW	SYST:CAL:ALL:IFBW	IFBW
Set the power level	SYST:CAL:ALL:PORT:SOUR:POW	PowerLevel
Set the power offset	SYST:CAL:ALL:PORT:SOUR:POW:OFFS	PowerOffset
Set the receiver atten	SYST:CAL:ALL:PORT:SOUR:POW:REC:ATT	ReceiverAttenuator
Set the source atten	SYST:CAL:ALL:PORT:SOUR:POW:ATT	SourceAttenuator
Set the User Calset Prefix	SYST:CAL:ALL:CSET:PREFix	UserCalsetPrefix
Set Path Configuration	SYST:CAL:ALL:PATH:CONF:ELEM	PathConfigurationElement
Read unique Cal properties	SYST:CAL:ALL:MCL:PROP:NAME:CAT?	PropertyNames
Read unique property values	SYST:CAL:ALL:MCL:PROP:VAL:CAT?	PropertyValues
Set property name/value	SYST:CAL:ALL:MCL:PROP:VAL	PropertyValue
Read Cal channel	SYST:CAL:ALL:GUID:CHAN?	N/A
Get GuidedCal handle	N/A	GuidedCalibration
For each channel, sets the ports to be calibrated.	SYST:CAL:ALL:CHAN:PORT	CalibrationPorts
Returns a final list of ports to be calibrated.	SYST:CAL:ALL:GUID:PORT?	SParameterCalPorts
Read generated Cal Sets	SYST:CAL:ALL:CSET:CATalog?	GeneratedCalsets

SMC Phase Reference Cal

SCPI

COM

Reset	SYST:CAL:ALL:PHAS:RES	Reset
Start Freq	SYST:CAL:PHAS:FREQ:STAR	StartFrequency
Stop Freq	SYST:CAL:PHAS:FREQ:STOP	StopFrequency
Read channel number	SYST:CAL:PHAS:GUID:CHAN?	N/A
Read all Phase Reference IDs	SYST:CAL:PHAS:REF:CAT?	GetConnectedPhaseReferences
Set Phase Reference	SYST:CAL:PHAS:REF	PhaseReference
Set Source Attenuator	SYST:CAL:PHAS:POW:ATT	SourceAttenuator
Set Cal Set name	N/A	CalSet
Perform Cal	Guided Cal commands	Guided Cal commands

Noise Cal using Power Meter

SCPI

COM

Select Receiver Cal method	SENS:NOIS:CAL:RMET	RcvCharMethod
----------------------------	--------------------	---------------

PMAR Cal and Zero	SCPI	COM
Set Zero method	SYST:CONF:EDEV:PMAR:ZERO	None
Perform Cal	SYST:CONF:EDEV:PMAR:CAL	None

Bal - Single-ended topology	SCPI	COM
Set Bal-SE measurement	CALC:FSIM:BAL:PAR:BALS	BalSMeasurement
Set Bal-SE topology (modified)	CALC:FSIM:BAL:DEvice	DUTTopology
Set port mapping	CALC:FSIM:BAL:TOP:BALS:PPORts	SetBSPorts
Read se port	CALC:FSIM:BAL:TOP:BALS:PPORts	BS_SEPort
Read Neg Bal port	CALC:FSIM:BAL:TOP:BALS:PPORts	BS_BalPortNegative
Read Pos Bal port	CALC:FSIM:BAL:TOP:BALS:PPORts	BS_BalPortPositive

The following are new programming commands for **PNA release A.09.50** See What's New

External DC Analyzer Control	SCPI	COM
Correction ON/OFF	SYST:CONF:EDEVice:DC:CORR	DCCorrection
Offset correction value.	SYST:CONF:EDEVice:DC:OFFS	DCOffset
Scale correction value.	SYST:CONF:EDEVice:DC:SCALE	DCScale
DC Type (Units).	SYST:CONF:EDEVice:DC:TYPE	DCType
Dwell Before/After Point	SYST:CONF:EDEVice:DC:DPOint	DwellPerPoint
Dwell Before Sweep value	SYST:CONF:EDEVice:DC:DSweep	SweepDwell
DC Source Sweep		
Source names catalog	SOURce:DC:Names	Sources
Enable source outputs	SOURce:DC:Enable	EnableAllOutput
Source state	SOURce:DC:State	State
Start DC	SOURce:DC:Start	Start
Stop DC	SOURce:DC:Stop	Stop
Data	SOURce:DC:Data	ListData

External Pulse Generator Control	SCPI	COM
PG Names catalog	SENSe:PULSe:CAT?	PulseGeneratorNames
Read the integer of the name	Not applicable	PulseGeneratorID
Set output channel	SYST:CONF:EDEVice:PULSe:CHAN	OutputChannel
Set output Hi amplitude (volts)	SYST:CONF:EDEVice:PULSe:HAMP	HighAmplitude
Set output Low amplitude (volts)	SYST:CONF:EDEVice:PULSe:LAMP	LowAmplitude
Set load impedance	SYST:CONF:EDEVice:PULSe:LIMP	LoadImpedance
Set source impedance	SYST:CONF:EDEVice:PULSe:SIMP	SourceImpedance
Master Mode	SYST:CONF:EDEVice:PULSe:MMODE	MasterMode
Optional Name/ID argument added to some Pulse gen commands.	Sens:Pulse	PulseGeneratorObject
Miscellaneous	SCPI	COM
Read configured device names	None	DeviceNames
Is a device name present?	None	IsDevicePresent
Move a trace to another window	DISP:WIND:TRAC:MOVE	Trace.Move
Recall softkey list sort preference	SYST:PREF:ITEM:MRU	RecallSoftkeysMostRecent
Default to "honor the channel <ch> argument in guided calibration SCPI commands."	SENS:CORR:COLL:GUID:CHAN:MODE	
Note: This MAY impact your legacy programs.		
Receiver Leveling frequency range	Sour:Pow:ALC:Rev:FTYPe	FrequencyType

The following are new programming commands for **PNA release A.09.42** See What's New

Misc Commands	SCPI	COM
Read CalSet stimulus values	None	GetErrorTermStimulus
Write CalSet stimulus values	None	PutErrorTermStimulus
Copy mechanical settings from specified channel	SENS:PATH:CONF:COPY	CopyFrom
Return a measurement handle of the trace object	None	Measurement
Read the PNA port which is connected to the DUT input.	None	DeviceInputPort
Read the PNA port which is connected to the DUT output.	None	DeviceOutputPort
Read the window number of the selected trace.	CALC:PAR:WNUM?	WindowNumber
Read the trace number of the selected trace.	CALC:PAR:TNUM?	N/A

Source / Role Commands (for apps)	SCPI	COM
Returns the roles for which sources can be used for the channel.	SENSe:ROLE:CATalog	DefinedRoles
Set and returns the source to be used in the specified role.	SENSe:ROLE:DEvice	RoleDevice

IMD and IM Spectrum Tone Power settings	SCPI	COM
Swept IMD Equal tone power	SENSe:IMD:TPOwer:EQualize:STATe	EqualTonePower
IM Spectrum Equal tone power	SENSe:IMD:TPOwer:SET	
Swept IMD Set power at DUT input or output	SENSe:IMS:TPOwer:EQualize:STATe	TonePowerSetAt
IM Spectrum set power at DUT input or output	SENSe:IMS:TPOwer:SET	

Pulse Commands	SCPI	COM
Set master pulse frequency	SENS:SWE:PULSe:MASTer:FREQ	MasterFrequency
Set master pulse period	SENS:SWE:PULSe:MASTer:PERiod	MasterPeriod
Set master pulse width	SENS:SWE:PULSe:MASTer:WIDTh	MasterWidth
Autoselect CW Sweep Time	SENS:SWE:PULSe:CWTime	AutoCWSweepTime

Capabilities Commands	SCPI	COM
Return parameters supported by the specified Measurement Class	None	SupportedParameters
Return list of supported IFBWs	None	IFBWList
Return list of ResBW's supported by IMSpectrum Apps	None	ResBWList

The following are new programming commands for **PNA release A.09.33** See What's New

FCA Commands		
	Sense:Correction:Guided	Not Applicable
Replaced SESSION commands .	Sense:Correction:Collect:Guided:SMC	
	Sense:Correction:Collect:Guided:VMC	

Multiple Power Sensors		
Enable multiple sensors	SENS:CORR:COLL:GUID:PSEN:MULT	UseMultipleSensors
Add sensors	SENS:CORR:COLL:GUID:PSEN:MULT:ADD	Add
Assign power sensor name	SENS:CORR:COLL:GUID:PSEN:MULT:NAME	Name
Remove sensors	SENS:CORR:COLL:GUID:PSEN:MULT:REM	Remove
Read the number of configured sensors	SENS:CORR:COLL:GUID:PSEN:MULT:COUNt?	Count
Set start freq	SENS:CORR:COLL:GUID:PSEN:MULT:FREQ:STAR	StartFrequency
Set stop freq	SENS:CORR:COLL:GUID:PSEN:MULT:FREQ:STOP	StopFrequency
Set connector type	SENS:CORR:COLL:GUID:PSEN:MULT:CONN	PowerSensorConnectorType
Set Cal Kit	SENS:CORR:COLL:GUID:PSEN:MULT:CKIT	PowerSensorCalKitType

Phase Control

Phase Sweep type	SENSe:SWEEp:TYPE	SweepType
Set fixed phase value	SOURce:PHASe [:FIXed]	FixedPhase
Phase sweep start value	SOURce:PHASe:STARt	StartPhase
Phase sweep stop value	SOURce:PHASe:STOP	StopPhase
Phase parameter	SOURce:PHASe:PARAmeter	PhaseParameter
Set Phase control mode	SOURce:PHASe:PARAmeter:MODE	PhaseControlMode
Set reference port	SOURce:PHASe:PARAmeter:PORT	PhaseReferencePort
Read available phase control modes for the port	SOURce:PHASe:PARAmeter:MODE:CAT?	PhaseParameterModes
Couple sweep settings	SOURce:PHASe:CONTRol:COUPlE	CouplePhasePortSettings
Set number of sweep iterations	SOURce:PHASe:CONTRol:ITERation	PhaseIterationNumber
Set sweep tolerance	SOURce:PHASe:CONTRol:TOLerance	PhaseTolerance
Set and read an array of phase offsets.	SOURce:PHASe:CORREction:DATA	PhaseCorrectionData
Use phase offset array.	SOURce:PHASe:CORREction:STATe	PhaseCorrectionEnabled
Set and read an array of ratioed power offsets.	SOURce:PHASe:POFFset:CORR:DATA	RatioedPowerCorrectionData
Use power offset array.	SOURce:PHASe:POFFset:CORR:STATe	RatioedPowerCorrectionEnabled
Set the fixed power ratioed value	SOUR:PHAS:POFFset:FIXed	FixedRatioedPower
Set the start power ratioed value.	SOUR:PHAS:POFFset:STARt	StartRatioedPower
Set the stop power ratioed value.	SOUR:PHAS:POFFset:STOP	StopRatioedPower

2 and 4-port Fixture De-embed

2-port Reverse	CALC:FSIM:SEND:DEEM:PORT<n>:SNP:REV	Reverse2PortAdapter
4-port remap	CALC:FSIM:EMB:NETW<n>:PMAP	NetworkPortMap
Extrapolate	CALC:FSIM:SNP:EXTR	EnableSnPDataExtrapolation
Extrapolate added to Cal Set De-embedding	CSET:FIXT:DEEM	Deembed
Extrapolate added to Cal Set Embedding	CSET:FIX:EMBED	Embed

Mixer Segment Sweep

Recalculate	SENS:MIX:RECalculate	ReCalculate
Segment Calculate	SENS:MIX:SEGM<n>:CALC	SegmentCalculate
Query Count	SENS:MIX:SEGM:COUNT?	SegmentCount
Add Segments	SENS:MIX:SEGM<n>:ADD	AddSegment
Delete Segments	SENS:MIX:SEGM<n>:DEL	DeleteSegment
Remove All Segments	SENS:MIX:SEGM:DEL:ALL	DeleteAllSegments
State	SENS:MIX:SEGM<n>[:STATe]	SegmentState
Number Of Points	SENS:MIX:SEGM<n>:POINTs	SegmentPoints
IFBW	SENS:MIX:SEGM<n>:BWIDth	SegmentIFBandwidth
Input Fixed Freq	SENS:MIX:SEGM<n>:INP:FREQ:FIXed	SegmentFixedFrequency
Input Start Freq	SENS:MIX:SEGM<n>:INP:FREQ:STARt	SegmentStartFrequency
Input Stop Freq	SENS:MIX:SEGM<n>:INP:FREQ:STOP	SegmentStopFrequency
Input Fixed/Swept	SENS:MIX:SEGM<n>:INP:FREQ:MODE	SegmentRangeMode
Input Power	SENS:MIX:SEGM<n>:INP:POWer	SegmentFixedPower
Output Fixed Freq	SENS:MIX:SEGM<n>:OUTP:FREQ:FIXed	SegmentFixedFrequency
Output Start Freq	SENS:MIX:SEGM<n>:OUTP:FREQ:STARt	SegmentStartFrequency
Output Stop Freq	SENS:MIX:SEGM<n>:OUTP:FREQ:STOP	SegmentStopFrequency
Output Fixed/Swept	SENS:MIX:SEGM<n>:OUTP:FREQ:MODE	SegmentRangeMode
Output (+/-)	SENS:MIX:SEGM<n>:OUTP:FREQ:SIDeband	SegmentMixingMode
Output Power	SENS:MIX:SEGM<n>:OUTP:POWer	SegmentFixedPower
LO Fixed Freq	SENS:MIX:SEGM<n>:LO<x>:FREQ:FIXed	SegmentFixedFrequency
LO Start Freq	SENS:MIX:SEGM<n>:LO<x>:FREQ:STARt	SegmentStartFrequency
LO Stop Freq	SENS:MIX:SEGM<n>:LO<x>:FREQ:STOP	SegmentStopFrequency
LO Fixed/Swept	SENS:MIX:SEGM<n>:LO<x>:FREQ:MODE	SegmentRangeMode
Input >LO	SENS:MIX:SEGM<n>:LO<x>:FREQ:ILTI	SegmentIsInputGreaterThanLO
LO Power	SENS:MIX:SEGM<n>:LO<x>:POWer	SegmentFixedPower
IF (+/-)	SENS:MIX:SEGM<n>:IF:FREQ:SIDeband	SegmentMixingMode

Miscellaneous

Read available Meas Classes	SYST:MCLass:CAT?	AvailableMeasurementClasses
Set receiver ratio to be used with receiver leveling.	SOURce:POWer:ALC:MODE:REC:RATio	ReceiverRatio
Removes adapters	SENS:CORR:COLL:GUID:ADAP:COUNT:ZERO	None
Perform Linear Interpolation	SYST:CORR:INT:LINear	None
Return if a CalSet exists	CSET:EXISTS?	Exists

The following are new programming commands for **PNA release A.09.30** See What's New

Marker Display

Readouts Per Trace	DISP:WIND:ANN:MARK:NUMB	MarkerReadoutsPerTrace
Stimulus decimal places	DISP:WIND:ANN:MARK:RES:STIM	MarkerReadoutStimulusPlaces
Response decimal places	DISP:WIND:ANN:MARK:RES:RESP	MarkerReadoutResponsePlaces
Readout position: X-axis	DISP:WIND:ANN:MARK:XPOS	MarkerReadoutXPosition
Readout position: Y-axis	DISP:WIND:ANN:MARK:YPOS	MarkerReadoutYPosition
Marker symbol	DISP:WIND:ANN:MARK:SYMB	MarkerSymbol

GCX - No new commands. [Learn more](#) .

Overload Preferences

Display receiver overload warnings?	SYST:PREF:ITEM:REC:CERR	ReportReceiverOverload
Turn source power OFF when a receiver is overloaded?	SYST:PREF:ITEM:REC:OVER:POW	RFOffOnReceiverOverload

Miscellaneous

Use Last Result for Source Power Cal	SOUR:POW:ALC:MODE:REC:LSPC	LastLevelingAsSPC
Selectively delete a channel	SYSTEM:CHANnels:DElete	Remove Method RemoveChannelNumber

Guided Power Cal

Perform Power Cal	SENS:CORR:COLL:GUID:PSEN:STATE	PerformPowerCalibration
Match-correction ON OFF	SENS:CORR:METHods:MATCh	MatchCorrectPower

Cal Set Items

Set or get name-value pair from calset	None	Item
Remove name-value pair from calset	None	RemoveItem
Enumerate name-value pair items in the calset.	None	EnumerateCalSets

The following are new programming commands for **PNA release A.09.22** See What's New

Returns the error correction state for the measurement.	CALC:CORR:INDicator?	ErrorCorrectionIndicator
Calibrating specific channels	SENS:CORR:COLL:GUID:CHAN:MODE	N/A

Port Mapping - Noise Figure Opt 028

Write port mapping	SENSe:NOISe:PMAP	SetPortMap
Read input port mapping	SENSe:NOISe:PMAP:INPut?	DeviceInputPort
Read output port mapping	SENSe:NOISe:PMAP:OUTPut?	DeviceOutputPort

The following are new programming commands for **PNA release A.09.20** See What's New

Configure Pulse Measurements

Pulse Meas Mode	SENS:SWE:PULS:MODE	PulseMeasMode
Autodetect Pulse mode	SENS:SWE:PULS:DETECTmode	AutoDetection
Set Pulse Mode (Narrow Wide)	SENS:SWE:PULS:WIDeband	WideBandDectionState
Autoselect IFBW	SENS:SWE:PULS:IFBW	AutoIFBandWidth
Autoselect IF Gain	SENS:SWE:PULS:IFGain	AutoIFGain
Autoselect Pulse clock period	SENS:SWE:PULS:PRF	AutoOptimizePRF
Autoselect Width and Delay	SENS:SWE:PULS:TIMing	AutoPulseTiming
Autoselect Pulse Gens	SENS:SWE:PULS:DRIVE	AutoSelectPulseGen

External Pulse Configuration

PulseSyncIn Trigger Polarity	Sense:Pulse:TPolarity	TriggerInPolarity
PulseSyncIn Trigger Type	Sense:Pulse:TType	TriggerInType

PSAT Marker Search

Initiate a PSAT search	CALC:MARK:PSAT:BACKoff	SearchPowerSaturation
Set and read PSAT backoff	CALC:MARK:PSAT:BACKoff	PMaxBackOff
Read PSat Out	CALC:MARK:PSAT:POUT?	POut
Read PSat In	CALC:MARK:PSAT:PIN?	Pin
Read PMax Out	CALC:MARK:PSAT:POUT:MAXimum?	PMaxOut
Read PMax In	CALC:MARK:PSAT:PIN:MAXimum?	PMaxIn
Read Gain Sat	CALC:MARK:PSAT:GAIN?	GainSaturation
Read Gain Max	CALC:MARK:PSAT:GAIN:MAXimum?	GainMax
Read Gain Linear	CALC:MARK:PSAT:GAIN:LINear?	GainLinear
Read Comp Sat	CALC:MARK:PSAT:COMP:SAT?	CompressionSaturation
Read Comp Max	CALC:MARK:PSAT:COMP:MAX?	CompressionMax

PNOP Marker Search

Initiate a PNOP search	CALC:MARK:PNOP:BACKoff	SearchPowerNormalOperatingPoint
Set and read PNOP backoff	CALC:MARK:PNOP:BACKoff	BackOff
Set and read PNOP Power Offset	CALC:MARK:PNOP:POFFset	PinOffset
Read Pnop Out	CALC:MARK:PNOP:POUT?	POut
Read Pnop in	CALC:MARK:PNOP:PIN?	Pin
Read Pnop Gain	CALC:MARK:PNOP:GAIN?	Gain
Read Pnop Comp	CALC:MARK:PNOP:COMPression?	Compression
Read PMax Out	CALC:MARK:PNOP:POUT:MAXimum?	PMaxOut
Read PMax In	CALC:MARK:PNOP:PIN:MAXimum?	PMaxIn
Read Gain Max	CALC:MARK:PNOP:GAIN:MAXimum?	GainMax
Read Comp Max	CALC:MARK:PNOP:COMPression:MAXimum?	CompressionMax
Read PBO Out	CALC:MARK:PNOP:BACKoff:POUT?	BackOffPout
Read PBO In	CALC:MARK:PNOP:BACKoff:PIN?	BackOffPin
Read PBO Gain	CALC:MARK:PNOP:BACKoff:GAIN?	BackOffGain

Include Phase in SMC measurements

Enable Phase	SENS:MIX:PHASe	EnablePhase
Set normalize point	SENS:MIX:NORMalize	NormalizePoint
Choose known delay or S2P	SENS:CORR:COLL:SESS:SMC:PHASe:METHod	DelayCalculationMethod
Set known delay	SENS:CORR:COLL:SESS:SMC:PHASe:DELay	FixedDelay
Set Cal Mixer Char S2P filename	SENS:CORR:COLL:SESS:SMC:PHASe:MIXer	MixerCharacterizationFile

Group Delay Aperture

Set aperture using frequency	CALC:GDELay:FREQ	Frequency
Set aperture using percent of span	CALC:GDELay:PERCent	Percent
Set aperture using fixed number of points	CALC:GDELay:POINts	Points
Set Preference to 2 points	SYST:PREF:ITEM:GDEL:TWOp	TwoPointGroupDelayAperture

Calibrations

Returns the names of the mechanical cal kits for unguided calibrations.	SENSe:CORR:COLL:CKIT:CAT?	CalKitTypes
Set/get by name which cal kit is currently selected for use by unguided cal.	SENSe:CORR:COLL:CKIT:PORT	CalKitType
Read pass fail status of tolerance limits on the target cal power	Not needed - read from SCPI error queue	LastCalPassedTolerance
Gets the power correction dB values from the prior iteration of the source power cal.	SOURce:POWer:CORR:DATA:PRIor	getSourcePowerCalDataEx getSourcePowerCalDataScalarEx

Fixturing

Compensate source power	CALC:FSIM:SEND:POW:PORT:COMP	EnablePowerCompensation
Create Cal Set with De-embedded fixture removed.	CSET:FIXTure:DEEMbed	None
Create Cal Set with Matching Network included.	CSET:FIXTure:EMBed	None

Noise Receiver (Opt 028)

Select Noise Receiver	SENSe:NOISe:RECeiver	NoiseReceiver
-----------------------	----------------------	---------------

Miscellaneous

Include SC12 Sweep	Sens:Mixer:Reverse	IncludeReverseSweep
Active Window Background	Disp:Color:ABACKground	ActiveBackground
Sweep Delay	SENS:SWEep:DWELl:SDELay	SweepDelay
Preference: On PRESET always turn power ON	SYST:PREF:ITEM:PRESet:POWer	PresetPowerState
Send and return Instrument state file to remote computer	SYSTem:SET	None
Read the display image into remote computer	HCOPy:SDUMp:DATA?	None
Set format of display image	HCOPy:SDUMp:DATA:FORM	None
Set and return LXI status	LXI:IDEN	LXIDeviceIDState
GCA Safe mode - max power	Sens:GCS:SAFE:MLIM	SafeSweepMaximumLimit
Set Line type of the window grid	DISP:WIND:TRAC:GRAT:GRID:LTYPE	GridLineType
Save Data	MMEM:STOR:DATA	SaveData

The following are new programming commands for **PNA release 9.10** See What's New

NFX

Sets the power sensor connector	SENS:CORR:COLL:GUID:PSEN:CONN	PowerSensorConnectorType
Sets the power sensor calkit	SENS:CORR:COLL:GUID:PSEN:CKIT	PowerSensorCalKitType
Sets power level for source power cal	SENS:CORR:COLL:GUID:PSEN:POW:LEV	PowerCalibrationLevel
Sets auto orientation state for noise tuner	SENS:NOISe:TUNer:ORient	AutoOrientTuner
Sets LO power calibration state	SENS:CORR:COLL:NOIS:LO:PCAL:STATE	EnableLOPowerCal
Sets the source pull technique to compute DUT S-parameters	SENS:NOISe:PULL	SourcePullForSParameters
Sets the state of ENR adapter de-embedding.	SENS:CORR:COLL:NOIS:ENR:ADAP:DEEM	ForceDeEmbedENRAdapter
Sets the state of Power Sensor adapter de-embedding.	SENS:CORR:COLL:NOIS:PSEN:ADAP:DEEM	ForceDeEmbedSensorAdapter

Help About commands

DSP Revision	SYST:CONF:REVISION:DSP?
DSP FPGA	SYST:CONF:REVISION:DSPFpga?
CPU Speed	SYST:CONF:REVISION:CPU?
Hostname	SYST:COMM:LAN:HOSTname?

Miscellaneous

Perform and apply Response (Normalization) cal	SENS:CORR:COLL:METHod	DoResponseCal
PulseSyncIn Trigger Polarity	Sense:Pulse:TPolarity	TriggerInPolarity
PulseSyncIn Trigger Type	Sense:Pulse:TType	TriggerInType

The following are new programming commands for **PNA release 9.0** See What's New

Configure External Devices

Adds an external device to the system.	SYST:CONF:EDEV:ADD	Add (External Device)
Returns names of all configured devices	SYST:CONF:EDEV:CAT?	Items
Set driver for the external device.	SYST:CONF:EDEV:DRIV	Driver
Set type of device.	SYST:CONF:EDEV:DTYP	DeviceType
Configuration path for external device.	SYST:CONF:EDEV:IOC	IOConfiguration
Enable or disable communication with device.	SYST:CONF:EDEV:IOEN	IOEnable
Activation state of the device.	SYST:CONF:EDEV:STAT	Active
Time out value for external device.	SYST:CONF:EDEV:TOUT	TimeOut

External Source Config

Set Dwell per Point	SYST:CONF:EDEV:SOUR:DPP	DwellPerPoint
Set Trigger Mode	SYST:CONF:EDEV:SOUR:TMOD	Trigger Mode
Set Trigger Port	SYST:CONF:EDEV:SOUR:TPORt	TriggerPort

Power Meter As Receiver (PMAR) Config

Enable min and max freqs	SYST:CONF:EDEV:PMAR:FLIM	LimitFrequency
Set Max freq	SYST:CONF:EDEV:PMAR:FMAX	MaximumFrequency
Set Min freq	SYST:CONF:EDEV:PMAR:FMIN	MinimumFrequency
Set max number of PM readings	SYST:CONF:EDEV:PMAR:READ:COUN	ReadingsPerPoint
Set tolerance level	SYST:CONF:EDEV:PMAR:READ:NTOL	ReadingsTolerance
Select sensor	SYST:CONF:EDEV:PMAR:SENS	SensorIndex
Set Cal Factor data	SYST:CONF:EDEV:PMAR:TABL:CFAC:DATA	ReferenceCalFactor
Set Cal Factor frequencies	SYST:CONF:EDEV:PMAR:TABL:CFAC:FREQ	Frequency
Set Power loss data	SYST:CONF:EDEV:PMAR:TABL:LOSS:DATA	Loss
Set Power loss frequencies	SYST:CONF:EDEV:PMAR:TABL:LOSS:FREQ	Frequency
Enable Power loss data	SYST:CONF:EDEV:PMAR:TABL:LOSS:STAT	UsePowerLossSegments
Set reference cal factor	SYST:CONF:EDEV:PMAR:TABL:RFAC	ReferenceCalFactor

Power Limit

Set power limit	SYST:POWer:LIMit	Limit
Power limit ON/OFF	SYST:POWer:LIMit:STATe	State
Power limit UI lock	SYST:POWer:LIMit:LOCK	Lock

Scale Coupling

Set method	DISP:WIND:TRAC:Y:COUP:METH	ScaleCouplingMethod
Enable window	DISP:WIND:TRAC:Y:COUP	ScaleCouplingState

Display and Print Colors

Set background color	DISP:COL:BACK	Background
Set labels and grid frame colors	DISP:COL:GRAT1	ActiveLabels
Set inner lines of all grids in all windows colors	DISP:COL:GRAT2	Grid
Set Inactive window label colors	DISP:COL:ILAB	InactiveLabels
Set limit line colors	DISP:COL:LIM1	FailedTraces
Set trace data and Limit Line colors	DISP:COL:TRAC:DATA	DataAndLimits
Set data trace marker colors	DISP:COL:TRAC:MARK	Markers
Set memory trace colors	DISP:COL:TRAC:MEM	Memory
Set memory trace marker colors	DISP:COL:TRAC:MMAR	MemoryMarkers
Load a color theme	DISP:COL:LOAD	LoadTheme
Saves the current color theme.	DISP:COL:STOR	StoreTheme
Resets to the default PNA colors.	DISP:COL:RES	ResetTheme

IMD and IMS Limited Port Mapping

Set port map	SENS:IMD:PMAP	SetPortMap
Read Input	SENS:IMS:PMAP SENS:IMD:PMAP:INP?	DeviceInputPort
Read Output	SENS:IMS:PMAP:INP? SENS:IMD:PMAP:OUTP? SENS:IMS:PMAP:OUTP?	DeviceOutputPort

ECal User Char to Disk (new and modified commands)

Returns the factory defined and user-defined connectors.	SENSe:CORR:CKIT:ECAL:CHAR:CONN:CAT?	None
Initiate a User Char - optionally check module memory.	SENSe:CORR:CKIT:ECAL:CHAR:INIT	InitializeEx
Saves a new characterization to disk memory	SENSe:CORR:CKIT:ECAL:CHAR:DMEM:SAVE	SaveToDiskMemory
Delete disk memory characterizations.	SENSe:CORR:CKIT:ECAL:DMEMory:CLear	None
Saves a disk memory characterization to an archive file.	SENS:CORR:CKIT:ECAL:EXPort	None
Imports the ECal characterization from the specified archive file.	SENSe:CORR:CKIT:ECAL:DMEMory:IMPort	None
Reads the user-characterization info from ECal module or PNA disk memory.	SENS:CORR:CKIT:ECAL:KNAM:INF?	None

Gain Compression Analysis

Enable a compression analysis trace	CALC:GCM:ANAL:ENAB	AnalysisEnable
Set CW frequency	CALC:GCM:ANAL:CWFR	AnalysisCWFreq
Set to discrete or interpolated CW frequencies	CALC:GCM:ANAL:ISD	AnalysisIsDiscreteFreq
Sets X-axis display	CALC:GCM:ANAL:XAX	AnalysisXAxis

Miscellaneous

Red Segment on Limit Line failures.	SYST:PREF:ITEM:RTOF	RedTraceOnFail
Returns the X-axis values for the selected trace.	CALC:X?	
Saves a Cal Kit to a file.	SENS:CORR:CKIT:EXP	
Returns the list of cal kits that use the specified connector.	SENSe:CORR:COLL:GUID:CKIT:CAT?	GetCompatibleCalKits
Gain Compression Saturation level	SENS:GCS:COMP:SAT:LEV	SaturationLevel
Set Cal FOM Range Preference	SENS:CORR:PREF:CAL:RANG	FrequencyOffsetRangeForCalComputations
Return the Calset X-axis FOM frequency range	SENS:CORR:CSET:STIMulus?	StimulusValues
SMC - perform separate power calcs	SENS:CORR:COLL:SESS:SMC:PWRC:SE P	SeparatePowerCal

The following are new programming commands for **PNA release 8.60** See What's New

Miscellaneous

Reads or writes the segment sweep table.	SENSe:SEGM:LIST	GetAllSegments
Optional arguments for...	Sens:Corr:Ckit:CLEar	
	Sens:Corr:Ckit:Init	

The following are new programming commands for **PNA release 8.55** See What's New

Miscellaneous

	SCPI	COM
Enable subpoint triggering	Sens:pulse:SUBPointTrig	SubPointTrigger

Description

IMD-X for Converters

Create or Change a Custom (Application) Measurement

Create a custom measurement

Change a custom measurement

Return handle to a converter object

Configure a Mixer

Calculate Input and Output frequencies

X-axis display

Discard Changes

Load a mixer setup

Save a mixer setup

Apply mixer settings

Assign a source to mixer input or LO.

Read all assigned roles

Read the source assigned to a role.

Set Input

Input start frequency

Input stop frequency

Input power level

Numerator - Input Frac.Mult

Denominator - Input Frac.Mult

Input to Swept or fixed

Input fixed frequency

Set LO

Recall a previously-configured external source. This is the only external LO configuration command.

LO freq sweep mode (fixed or swept)

LO fixed frequency

LO start frequency

LO stop frequency

LO power

LO power start

LO power stop

Numerator - LO Frac.Mult

Denominator - LO Frac.Mult

Input Greater / Less than LO

Set IF

Sideband (high or low)

Set Output

Sideband (high or low)

Output start frequency

Output stop frequency

Output to swept or fixed

Output fixed frequency

IMD_x Cal

Enable LO Power cal

The following are new programming commands for **PNA release 8.50** See What's New

Compression Marker

Compression Marker level found.	CALC:MARKer:COMPression:LEVel	CompressionLevel
Read Compression Marker Input power	CALC:MARKer:COMPression:PIN	CompressionPin
Read Compression Marker Output power	CALC:MARKer:COMPression:POUT	CompressionPout
New Search function	CALC:MARKer:FUNCTion:SEL	SearchCompressionPoint
New Execute function	CALC:MARKer:FUNCTion:EXEC	SearchCompressionPoint

Port Extensions

Port Ext in distance	SENS:CORR:EXT:PORT:DIST	PortDistance
Set distance units	SENS:CORR:EXT:PORT:UNIT	PortDistanceUnit
Set Media per port	SENS:CORR:EXT:PORT:MEDIum	PortMedium
Set waveguide cutoff freq per port	SENS:CORR:EXT:PORT:WGCutoff	PortWGCutoffFreq
Set Velocity Factor per port	SENS:CORR:EXT:PORT:VELF	PortVelocityFactor
Couple to system Velocity Factor	SENS:CORR:EXT:PORT:SYSV	PortCoupleToSystemVelocity
Couple to system Media type	SENS:CORR:EXT:PORT:SYSM	PortCoupleToSystemMedia

Electrical Delay

Delay in distance	CALC:CORR:EDELay:DISTance	ElecDistanceDelay
Set units for distance	CALC:CORR:EDELay:UNIT	ElecDistanceDelayUnit

Phase Sweep

Enable Phase sweep	CALC:FSIM:BAL:PHAS:SWE:STAT	PhaseSwpState
Start Phase port 1	CALC:FSIM:BAL:BPOR:SWE:PHAS:STAR	BalPort1StartPhase
Start Phase port 2	CALC:FSIM:BAL:BPOR:SWE:PHAS:STAR	BalPort2StartPhase
Stop Phase port 1	CALC:FSIM:BAL:BPOR:SWE:PHAS:STOP	BalPort1StopPhase
Stop Phase port 2	CALC:FSIM:BAL:BPOR:SWE:PHAS:STOP	BalPort2StopPhase
Enable as fixture offset	CALC:FSIM:BAL:FIXT:SWE:PHAS	PhaseSwpAsFixture

Other

ASYN argument added to Cal ACQUIRE commands	Learn more	Not Applicable
Returns ECal orientation.	SENS:CORR:CKIT:ECAL:ORI?	None

The following are new programming commands for **PNA release 8.35** See What's New

FIFO Data Buffer

FIFO ON OFF	SYST:FIFO[:STATE]	State
Read number of data points	SYST:FIFO:DATA:COUNt?	DataCount
Read data	SYST:FIFO:DATA?	Data
Read data compact form	None	DataInCompactForm
Clear data	SYST:FIFO:DATA:CLEar	Clear

Other N5264A Commands

FastCW	SENS:SWE:TYPE:FACW	FastCWPointCount
Enable Point Averaging	SENS:AVER:MODE	AverageMode
Enable Point Sweep	SENS:SWE:GEN:POINtsweep	PointSweepState
Set Trace Sweep	SENS:SWE:TRIG:MODE	Trigger Mode

The following are new programming commands for **PNA release 8.33** See What's New

Miscellaneous

Set display format units	Calc:Format:Unit	FormatUnit
Perform trace max	Disp:TMAX	TraceMax
Fast sweep mode	SENS:SWE:SPE	SweepSpeedMode
Launch Cal Wizard for apps (new behavior)	SYST:CORR:WIZ	LaunchCalWizard
Queries the TCP/IP port number for a TCP/IP socket connection.	SYST:COMM:TCPIP:CONT	None
Set CWFreq to Marker location	CALC:MARK:SET	SetCWFreq
Returns a list of channel numbers	SYST:CHAN:CAT?	Not new
Returns measurement numbers	SYST:MEAS:CAT?	Not new
Returns trace number	SYST:MEAS<n>:TRACe?	Not new
Returns Meas name	SYST:MEAS:NAME?	Not new
Return window number	SYST:MEAS<n>:WINDow?	Not new
Returns window numbers	SYST:WIND:CAT?	Not new
Same as calc:par:sel except takes a meas number	CALC:PAR:MNUM:[SEL]	Not new
Returns Limit line pass/fail status	CALC:LIMIT:FAIL?	Not new
Deletes the current limit line?	CALC:LIMIT:DATA:DELete	Not new

Swept IMD

Create a measurement	Calc:Custom:Define	CreateCustomMeasurementEx
Set sweep type	SENS:IMD:SWE:TYPE	SweepType
Set DeltaF	SENS:IMD:DFR	DeltaFrequency
Set center freq	SENS:IMD:FREQ:FCEN	FrequencyCenter
Start for center freq sweep	SENS:IMD:FREQ:FCEN:STAR	FrequencyCenterStart
Stop for center freq sweep	SENS:IMD:FREQ:FCEN:STOP	FrequencyCenterStop
Center for center freq sweep	SENS:IMD:FREQ:FCEN:CENT	FrequencyCenterCenter
Span for center freq sweep	SENS:IMD:FREQ:FCEN:SPAN	FrequencyCenterSpan

Start for DeltaF sweep	SENS:IMD:FREQ:DFR:STAR	DeltaFrequencyStart
Stop for DeltaF sweep	SENS:IMD:FREQ:DFR:STOP	DeltaFrequencyStop
Set F1 for CW and Power sweep	SENS:IMD:FREQ:F1	F1Frequency
Set F2 for CW and Power sweep	SENS:IMD:FREQ:F2	F2Frequency
Set main tone IFBW	SENS:IMD:IFBW:MAIN	MainToneIFBandwidth
Set product tones IFBW	SENS:IMD:IFBW:IMT	IMToneIFBandwidth
Enables power coupling for F1 and F2	SENS:IMD:TPOW:COUP	CoupleTonePower
Set power level for F1 tone	SENS:IMD:TPOW:F1	TonePower
Set power level for F2 tone	SENS:IMD:TPOW:F2	TonePower
F1 start for power sweep	SENS:IMD:TPOW:F1:STAR	TonePowerStart
F1 stop for power sweep	SENS:IMD:TPOW:F1:STOP	TonePowerStop
F2 start for power sweep	SENS:IMD:TPOW:F2:STAR	TonePowerStart
F2 stop for power sweep	SENS:IMD:TPOW:F2:STOP	TonePowerStop
Read highest product allowed	None	HighestOrderProduct

For CTB, CSO, and XMod parameters

Normalization Mode	SENS:IMD:NORM:MODE	CompositeNormalizationMode
Normalized CSO power	SENS:IMD:CSO:NORM:POW	CompositeNormalizedCSOPower
CSO Offset	SENS:IMD:CSO:OFFS	CSOOffset
CSO Number of Distortion products	SENS:IMD:CSO:NDPR	CSONumDistortionProducts
Normalized CTB power	SENS:IMD:CTB:NORM:POW	CompositeNormalizedCTBPower
CTB and XMod Number of carriers	SENS:IMD:CTB:NCAR	CTBXMODNumCarriers
CTB Offset	SENS:IMD:CTB:OFFS	CTBOffset

IMD Calibration

Set Cal frequencies	SENS:CORR:IMD:CAL:FREQ	CalibrationFrequencies
Max Products	SENS:CORR:IMD:MPR	MaxProduct
Set power	SENS:CORR:IMD:POW	PowerLevel
Sensor Cal Kit	SENS:CORR:IMD:SENS:CKIT	PowerSensorCalKitType
Sensor connector	SENS:CORR:IMD:SENS:CONN	PowerSensorConnectorType

The following are new programming commands for **PNA release 8.2** See What's New

iTMSA

Set Stimulus Mode	CALC: FSIM:BAL:STIM:MOD	Mode
		BalPort1PhaseOffset
Set Phase Offset	CALC: FSIM:BAL:BPOR:OFFS:PHAS	BalPort2PhaseOffset
Set Phase Offset as fixture	CALC: FSIM:BAL:FIXT:OFFS:PHAS	PhaseAsFixture
		BalPort1PowerOffset
Set Power Offset	CALC: FSIM:BAL:BPOR:OFFS:POW	BalPort2PowerOffset
Set Power Offset as fixture	CALC: FSIM:BAL:FIXT:OFFS:POW	PowerAsFixture
Set Source power for balanced ports	SOUR:POW	TestPortPower
Returns the number of source ports.	N/A	chan.SourcePortCount
Returns the string names of source ports.	N/A	chan.SourcePortNames
Returns the port number for the specified string port name.	N/A	chan.GetPortNumber

Uncoupled Power Sweep

Set Start power for uncoupled power sweep	SOUR:POW:PORT:START	StartPowerEx
Set Start power for uncoupled power sweep	SOUR:POW:PORT:STOP	StopPowerEx

Choose FCA ports

Map PNA to DUT ports	SENS:MIX:PMAP	SetDutPorts
Read Input port number	SENS:MIX:PMAP:INP?	DeviceInputPort
Read Output port number	SENS:MIX:PMAP:OUTP?	DeviceOutputPort

LXI

Returns Structured status of the PNA networking configuration.	None	GetIPConfigurationStruct
Returns string status of the PNA networking configuration.	None	LANConfiguration
Resets the PNA LAN configuration.	None	LANConfigurationInitialize
Modifies settings of the PNA computer networking configuration.	None	SetIPConfiguration
Displays the LAN Status dialog with LAN Status Indicator showing IDENTIFY.	None	LXIDeviceIDState

Miscellaneous

Reset Preference Defaults	SYST:PREF:DEF	cap.RestoreDefaults
Returns the Measurement Class name	Sens:Class:Name?	Get_MeasurementClass
GCA - Returns number of iterations required in a SMART Sweep	CALC:GCDAT:ITER	TotalIterations

The following are new programming commands for **PNA release 8.0** See What's New

Gain Compression Setup

Number of frequency points	SENS:GCS:SWE:FREQ:POIN	NumberOfFrequencyPoints
Number of power points	SENS:GCS:SWE:POW:POIN	NumberOfPowerPoints
Maximum number of points	None	MaximumNumberOfPoints
Total number of points	None	TotalNumberOfPoints
Acquisition mode	SENS:GCS:AMOD	AcquisitionMode
Smart tolerance	SENS:GCS:SMAR:TOL	SmartSweepTolerance
Smart Iterations	SENS:GCS:SMAR:MIT	SmartSweepMaximumIterations
Smart settling time	SENS:GCS:SMAR:STIM	SmartSweepSettlingTime
Smart show iterations	SENS:GCS:SMAR:SIT	SmartSweepShowIterations
Read compression failures	SENS:GCS:SFA?	SearchFailures
Write port map	SENS:GCS:PORTM	SetPortMap
Read Port Map (Input)	SENS:GCS:PORT	DeviceInputPort
Read Port Map (Output)	SENS:GCS:PORT	DeviceOutputPort
End of Sweep	SENS:GCS:EOS	EndOfSweepOperation
Linear input power	SENS:GCS:POW:LIN:INP:LEV	InputLinearPowerLevel
Reverse Power	SENS:GCS:POW:REV:LEV	ReverseLinearPowerLevel
Start power	SENS:GCS:POW:STAR:LEV	chan.Start Power
Stop power	SENS:GCS:POW:STOP:LEV	chan.Stop Power
Compression algorithm	SENS:GCS:COMP:ALG	CompressionAlgorithm
Compression Level	SENS:GCS:COMP:LEV	CompressionLevel
Backoff Level	SENS:GCS:COMP:BACK:LEV	CompressionBackoff
X Delta	SENS:GCS:COMP:DELT:X	CompressionDeltaX
Y Delta	SENS:GCS:COMP:DELT:Y	CompressionDeltaY
Interpolation	SENS:GCS:COMP:INT	CompressionInterpolation
Safe Sweep enable	SENS:GCS:SAFE:ENAB	SafeSweepEnable
Safe Sweep coarse	SENS:GCS:SAFE:CPAD	SafeSweepCoarsePowerAdjustment
Safe Sweep fine	SENS:GCS:SAFE:FPAD	SafeSweepFinePowerAdjustment
Safe Sweep threshold	SENS:GCS:SAFE:FTHR	SafeSweepFineThreshold
Read all GCA data	CALC:GCDData:DATA	GetRaw2DData
Read real GCA data	CALC:GCDData:REAL	GetDataIm
Read imaginary GCA data	CALC:GCDData:IMAG	GetDataRe

Noise Figure Setup

Create Noise figure meas	Calc:Cust:Def	CreateCustomMeasurementEx
Sets the number of impedance states to use	SENS:NOIS:IMP:COUN	ImpedanceStates
Noise averaging ON and OFF	SENS:NOIS:AVER:STAT	NoiseAverageState
Set averaging of noise receiver.	SENS:NOIS:AVER	NoiseAverageFactor
Set bandwidth of noise receiver.	SENS:NOIS:BWID	NoiseBandwidth
Set gain state of noise receiver.	SENS:NOIS:GAIN	NoiseGain
Sets noise tuner identifier	SENS:NOIS:TUN:ID	NoiseTuner
Sets the port identifier of the ECal noise tuner that is connected to the PNA Source.	SENS:NOIS:TUN:INP	NoiseTunerIn
Sets the port identifier of the ECal noise tuner that is connected to the DUT.	SENS:NOIS:TUN:OUTP	NoiseTunerOut
Set the excess noise source ON or OFF.	CONTrol:NOISe:SOURce or OUTPut:MANual:NOISe[:STATe]	NoiseSourceState
Set mechanical switches	SENS:PATH:CONF:ELEM	PathConfiguration
Sets the default setting for the Noise Tuner switch.	SYST:PREF:ITEM:SWIT:DEF	Port1NoiseTunerSwitchPresetsToExternal

Noise Figure Cal

Create Noise Cal object	N/A	CreateCustomCalEx
Set Noise Calibration method	SENS:NOIS:CAL:METH	CalMethod
Noise source ENR filename	SENS:NOIS:ENR:FIL	ENRFile
Set noise source Cal Kit type	SENS:NOIS:SOUR:CKIT	NoiseSourceCalKitType
Set ambient temperature	SENS:NOIS:TEMP:AMB	AmbientTemperature
Sets noise source connector type	SENS:NOIS:SOUR:CONN	NoiseSourceConnectorType
Set Noise source temperature	SENS:CORR:TCOL:USER:VAL	NoiseSourceCold

Noise Figure ENR File Data Management

Set ENR calibration data.	SENS:CORR:ENR:CAL:TABL:DATA	PutENRData
Read ENR calibration data.	SENS:CORR:ENR:CAL:TABL:DATA?	GetENRData
Get/set ID of ENR table.	SENS:CORR:ENR:CAL:TABL:ID:DATA	ENRID
Get/set serial number of noise source.	SENS:CORR:ENR:CAL:TABL:SERial:DATA	ENRSN
Load ENR table from file.	MMEMory:LOAD:ENR	LoadENRFile
Save ENR table to file.	MMEMory:STORe:ENR	SaveENRFile

Custom Cal Window

Turn ON OFF Custom Cal window.	SENS:CORR:COLL:DISP:WIND	DisplayNAWindowDuringCalAcquisition
Show NO Custom Cal windows.	SENS:CORR:COLL:DISP:WIND:AOFF	DisplayOnlyCalWindowDuringCalAcquisition
Specify channel to sweep before Cal acquisition.	SENS:CORR:COLL:SWE:CHAN	AllowChannelToSweepDuringCalAcquisition
Sweep NO channel before Cal acquisition.	SENS:CORR:COLL:SWE:CHAN:AOFF	SweepOnlyCalChannelDuringCalAcquisition
Preview sweep before remote Cal acquisition.	SENS:CORR:COLL:GUID:PACQuire	SetupMeasurementsForStep

Miscellaneous

Set Trigger sweep mode	SENS:SWE:TRIG:MODE	Trigger Mode
Copy a Cal Set	SENSe:CORR:CSET:COPY	

COM versus SCPI

There are two methods you can use to remotely control the PNA: COM and SCPI. The following topics can help you choose the method that best meets your needs:

- [Software Connection](#)
- [Physical Connection](#)
- [Programming Languages](#)

Other Topics about COM Concepts

Software Connection

COM uses a binary protocol, allowing you to directly invoke a PNA feature. This is more efficient than SCPI. For example, the following statement calls directly into the PNA, executing the routine GetIDString.

```
PNA.GetIDString()
```

SCPI is a text based instrument language. To retrieve the ID string, you would send the following text string to the PNA:

```
IbWrite( "*IDN?")
```

The PNA SCPI parser would first decode this text string to determine that the user has asked for the PNA to identify itself. Then the parser would call the COM method GetIDString().

The Physical Connection

Internal Control

With either COM or SCPI, the best throughput is attained by using the PNA's internal PC to execute your test code. However, if your test code uses too much system resources (CPU cycles and/or memory), this will slow the PNA's performance.

Using the SICL I/O Libraries, you can also connect to the PNA from a program running on the PNA.

External Control

You can control the PNA from a remote PC using either COM or SCPI.

COM - (Component Object Model) can be used to access any program like the PNA (835x.exe) or

library (.dll) that exposes its features using a COM compliant object model. These programs or libraries are called "servers". Programs (like your remote program on your PC) that connect to and use the features of these servers are called "clients."

With COM, the server and the client do not need to reside on the same machine. DCOM, or distributed COM, makes the location of the server transparent to the client. When you access the PNA from a remote computer, you are using DCOM. In this case, the mechanical transport is a LAN (local area network).

However, using COM can add additional complexity:

- There are some DCOM security issues that may be a problem for you. [Learn more.](#)
- Using the default interface when compiling type libraries results in code that will only run with the latest firmware. [Learn more.](#)

SCPI - Using a GPIB interface card in a remote computer, you can connect to the instrument using a GPIB cable. There are some constraints on the length of this cable and the number of instruments that can be daisy-chained together.

Using the Agilent SICL I/O libraries, you can connect to the instrument over a LAN connection.

(LAN or INTERNAL) You can send SCPI commands using COM with the [ScpiStringParser](#) object.

If you have legacy code written in SCPI for another network analyzer, you may be able to leverage that code to control the PNA. However, the PNA uses a different platform than previous Agilent Network Analyzers. Therefore, not all commands have a direct replacement. See the PNA [Code Translator Application](#).

Programming Languages

You can program the PNA with either COM or SCPI using several languages. The most common include:

Agilent VEE - With this language you can send text based SCPI commands and also use automation. VEE 6.0 or later is recommended.

Visual Basic - This language has great support for automation objects and can be used to drive SCPI commands. The use of VISA drivers for your GPIB hardware interface will make the task of sending SCPI commands easier.

C++ - This language can do it all. It is not as easy to use as the above two, but more flexible.

24-Mar-2009 Updated

Remotely Specifying a Source Port

In the 'not-too-distant past', it was a simple task to specify a PNA source port. It was either port 1 or port 2. Now, for the following reasons, it is not so simple:

- **Internal 2nd sources** are now offered on various PNA models. However, some source ports do not have a port number. One example is the second source on the PNA-X 2-port model (option 224). [Learn more about Internal Second Sources](#).
- **External sources** can now be controlled by the PNA as though they are internal sources. External sources do not have a source port number, but use String names as identifiers.
 - **For FCA ONLY:** Once configured using the [Configuration dialog](#), an external source can be selected remotely and controlled by the PNA by specifying the LOName using [SCPI](#) or [COM](#).
 - **All other uses for External sources:** The external source must be configured and selected from the [External Source](#) dialog. You can then save an [Instrument State file](#), then recall that state file remotely.
- **Multiport test sets**...choose between ports 1 through port N, where N is the number of ports on the test set. You still use a port number, but this port number refers to a logical port. The Port mapping feature maps the logical port to a physical port. [Learn more about Multiport test sets](#).
- **iTMSA (Opt 460)** When this option is present, the string names for balanced source ports are returned with the appropriate COM and SCPI commands. For example, "SE Port 1" is used to access 'Single-ended Port 1'.

Source Port String Names

The PNA User Interface (UI) makes it easy to configure and select the sources and ports. Remotely however, string names are used now, in addition to port numbers, to specify a Source port.

COM - The existing COM commands specify source ports as numbers and they are still used. It is necessary to learn the port number from the string using the [GetPortNumber Method](#). Port numbers are assigned dynamically depending on whether [external sources are selected](#) and the number of ports of the PNA.

- [SourcePortNames Property](#)
- [GetPortNumber Method](#)
- [SourcePortCount Property](#).

An example:

```
dim app
set app = CreateObject("Agilentpna835x.application")
dim channel
```

```
set channel = app.Channel
dim portnum
portnum = Channel.GetPortNumber("Src2 Out1")
app.CreateMeasurement 1,"A",portnum
```

SCPI - ALL of the existing SCPI commands that specify a source port are extended to also allow the source port to be specified using string names. For example, send the following command to set the power on Src2 Out1:

- [SOUR:POW 5, "Src2 Out1"](#)
- Use [Source:Cat?](#) to list the available source port string names.

Last Modified:

23-May-2008	Added iTMSA option
24-Apr-2008	Several edits
5-Nov-2007	MX New topic

Shut Down the PNA Remotely

Shutdown the PNA remotely by creating and then executing a macro.

Learn more about [Macros in the PNA](#)

Create the Macro

1. Click **Utility**, then **Macro**, then **Macro Setup**
2. In the Macro Setup dialog, select a blank line, then click **Edit**.
3. In the Edit Macro dialog:
 - a. Under Title, enter **Shutdown**
 - b. Under Macro Executable, enter **Shutdown**
 - c. Under Macro run string parameters, enter **-s -t 0** This will wait 0 seconds before executing the command.

Execute the Macro Remotely

Send the following SCPI command:

- [SYST:SHOR<n>:EXEC](#) where <n> is the number (formerly 'blank' line) of the macro that is stored in the PNA.

Variations

To see variations on this command, on any Windows computer:

- From any command prompt, enter **help shutdown**

Last Modified:

31-Jul-2013 Modified for Win 7
22-May-2009 New topic

Fast Sweep (Opt 118) and other Antenna Features

The following features, used with the PNA-X models and the N5264A, were designed specifically for Antenna applications.

- **Fast Sweep Features (Opt 118)**
 - [FIFO Buffer](#) and [Fast CW](#)
 - [FIFO Buffer and Fast Groups](#)
 - [FIFO Buffer and Fast Segments](#)
 - **Other Useful Antenna Features**
 - [Point Averaging](#)
 - [Point Sweep](#)
 - [Trace Triggering](#)
 - **See Also**
 - [N5264A](#)
 - [Pulsed Measurements](#)
 - [Frequency \(Security\) Blanking](#)
 - [External Triggering](#)
-

Fast Sweep Features (Opt 118)

The following Fast Sweep features allow you to **very quickly** measure and download data to a remote computer.

- [Fast CW](#), [Fast Groups](#), and [Fast CW Segments](#) all work ONLY with the FIFO Data Buffer.
- These features can be used ONLY with [SCPI or COM commands](#). COM is faster than SCPI when using [DataInCompactForm](#). Otherwise, SCPI is faster than COM.

The Fast Sweep features are demonstrated in the following example program:

Download and install the **FIFO Tester** - a C# program that uses either the COM or SCPI interface to download data to, and read data from, the FIFO buffer. Source code is also provided. See <http://na.tm.agilent.com/pna/apps/applications.htm>

The FIFO Data Buffer

The FIFO (First-IN, First-OUT) data buffer is a circular buffer that allows very fast Read-Write access.

- When enabled, all the data gathered is placed into a 4 GB FIFO buffer.
- You can write to, and simultaneously read from, the FIFO buffer.
- A maximum of 1 million data points can be read for each query.
- REAL / IMAGINARY pairs is the ONLY supported format for the FIFO buffer.
- A preset or instrument state recall will turn off the FIFO buffer collection.
- When more than one measurement is present, data from each measurement is stored in the FIFO buffer in the following order. These measurements are separated into lines for easier reading.

4-port models

R, A, B, C, D,

R/R, A/R, B/R, C/R, D/R,

R/A, A/A, B/A, C/A, D/A,

R/B, A/B, B/B, C/B, D/B,

R/C, A/C, B/C, C/C, D/C,

R/D, A/D, B/D, C/D, D/D

2-port models

R1, R2, A, B,

R1/R1, R2/R1, A/R1, B/R1

R1/R2, R2/R2, A/R2, B/R2

R1/A, R2/A, A/A, B/A

R1/B, R2/B, A/B, B/B

S-parameters are pre-defined, ratioed-receiver measurements. [Learn more](#). S-parameters are placed in the FIFO in order based on their underlying receivers. For example, S21 is placed into the FIFO in the same manner as B/R1.

Fast CW

In Fast CW mode the PNA display is not updated. There is no background computation or other 'interference' from the PNA computer. Therefore, data is acquired real-time.

The following **requirements** must be met **before** sending the Fast CW command.

- FIFO is ON
- A single channel is being measured. Other channels can be in Hold.
- All measurements are acquired in a single sweep.

IMPORTANT - Fast CW and IF Bandwidth setting

- IF Bandwidth of **10 kHz and lower** - Data is transferred immediately to the FIFO after every acquisition.
- IF Bandwidths **greater than 10 kHz** - Data is transferred to the FIFO in groups. A triggered acquisition is NOT placed into the FIFO buffer until either the total number of points is completed, or an intermediate group of points is finished. The number of points within a group differs for each IF Bandwidth setting.

Notes:

- See example programs in [SCPI](#) and [COM](#).
- Fast CW sets the number of data points, overwriting the standard channel setting.
- When exiting Fast CW, the FIFO data buffer is cleared.
- External trigger signals are allowed only through the rear-panel [Trig In connector](#) - NOT the Aux1 and Aux2 In connectors.
- An error message appears if triggering is sent to the PNA faster than it can respond.

Fast Groups with FIFO Data Buffer

With this speed optimization feature, interaction with Windows or other PNA 'overhead' calls are suspended, allowing very fast and predictable measurement timing.

Fast Groups is automatically enabled when the following **requirements** are met:

- FIFO is ON

- A single channel is being measured. Other channels can be in Hold.
- All measurements are acquired in a single sweep.
- [Group trigger is enabled](#) with count > 1.

Notes:

- Fast CW can **NOT** be used with Fast Groups.
- Fast Groups and Fast CW Segments were designed to be used together, but not required.
- The [FIFO Tester](#) example program demonstrates this feature.

Fast CW Segments with FIFO Buffer

In this optimization feature, each CW segment (where the start and stop frequency is identical) within a channel is measured at speeds as fast as the Fast CW mode sweep.

Fast CW Segments is automatically enabled when the following **requirements** are met:

- FIFO is ON
- Start and stop frequency of a segment is identical.
- External trigger signals are allowed only through the rear-panel [Trig In connector](#) - NOT the Aux1 and Aux2 In connectors.

Notes:

- Fast CW can **NOT** be used with Fast CW Segments.
- The sweep can include non-CW segments, but these are not acquired in Fast mode.
- Fast Groups and Fast CW Segments were designed to be used together, but not required.
- The [FIFO Tester](#) example program demonstrates this feature.
- In Fast CW Segments, when data is not being acquired in real-time, the following message appears:
Caution: Sweep time jitter. Try reducing the number of segments. To avoid this error, reduce the number of segments in the channel.

Other Antenna Features

Point Averaging

This feature is selected on the [Average](#) dialog.

When selected, each data point is measured the specified number of averages before stepping to the

next data point. When [point trigger](#) is selected, only one trigger is required for each data point regardless of the number of averages.

Point Sweep

This feature is selected on the [Sweep Setup](#) dialog.

In Point Sweep mode, the PNA measures both the forward and reverse parameters at each frequency point before stepping to the next frequency. The display trace is updated as each data point is measured. Point sweep is the same as stepped sweep mode of the 8510 and 8530.

Trace Triggering

This feature is selected under [Trigger Mode](#) on the Trigger dialog.

Available ONLY when [Point Sweep](#) is selected. Each trigger signal causes two identical measurements to be triggered separately - one trigger signal is required for each measurement. Other trigger mode settings cause two identical parameters to be measured simultaneously.

Trace triggering is NOT permitted when a channel is using a 2 port (or more) S-Parameter calibration.

See Also

- [Pulsed Measurements](#)
- [Frequency \(Security\) Blanking](#)
- [External Triggering](#)

Last Modified:

8-Mar-2013	Added 2-port FIFO order
5-Jan-2011	Added Real / Imag pairs format ONLY
21-Jun-2010	Added important note
6-Mar-2009	MX New topic

Frequency Converter Application (Option 083)

FCA includes both Scalar (SMC) and Vector (VMC) measurements and calibrations.

In this topic:

- [What's New in FCA](#)
- [FCA Options Explained](#)
- [Comparison of VMC and SMC](#)
- [Requirements and Limitations](#)
- [How to make SMC or VMC Measurements](#)
 - [Create a Measurement](#)
 - [Make Measurement Settings](#)
 - [Sweep Tab](#)
 - [Segment Sweep](#)
 - [Power Tab](#)
 - [Mixer Setup Tabs](#) (separate topic)
 - [Select X-axis Display](#)
 - [Save Trace Data](#)
 - [Avoid Spurs](#)

See Also

- [SMC Measurements and Calibrations](#)
 - [SMC + Phase](#)
- [VMC Measurements and Calibrations](#)
- [Configure an External LO Source](#)
- [SMC with a Booster Amp](#)
- [Characterize Adaptor Macro](#)

- [Measure a DUT with an Embedded LO](#)
- For a detailed understanding of FCA, see our [Mixer Measurements App Notes](#).

Examples

- [How to make a VMC Measurement](#)
- [How to make an SMC Measurement](#)

Other Application topics

What's new in FCA

Rev A.09.33

- [New configuration dialogs](#)
- [Segment Sweep](#)
- [Power Sweep](#)
- [Save CSV Trace Data](#)

FCA Options Explained

- Option 083 provides FCA which includes [Scalar Mixer \(SMC\)](#) and [Vector Mixer \(VMC\)](#) Measurements.
- Option 082 provides ONLY SMC measurements. This is the ONLY FCA option that is allowed on the N5230C.
- Option 084 provides [Embedded LO](#) measurements. This option requires one of the [Converter Applications](#).
- [See all PNA-X Options and Configurations](#).

Comparison of SMC and VMC

	Scalar Mixer Calibration	Vector Mixer Calibration
	See Hardware setup	See Hardware setup
Overview	<p>Provides highest Scalar accuracy for measurements of conversion loss/gain.</p> <p>Optionally measures phase</p> <p>Combines SOLT and power-meter calibration.</p> <p>Simpler setup than Vector Mixer Calibration.</p>	<p>Provides unparalleled accuracy for measurements of relative phase and absolute group delay.</p> <p>Uses combination of SOLT standards and a reciprocal mixer/filter pair during calibration.</p> <p>More complicated setup and calibration procedure than Scalar Mixer Calibration.</p> <p>After calibration, both reciprocal and non-reciprocal mixers and converters can easily be measured.</p>
Measurements Offered	<p>Both forward and reverse directions.</p> <p>DUT can be connected to any PNA ports.</p>	<p>Amplitude response VC_{21}</p> <p>Phase response</p> <p>Group delay</p> <p>DUT input must be connected to PNA port 1.</p> <p>DUT output can be connected to any other PNA port.</p>
Equipment Required	Power meter and sensor	<p>Reference mixer</p> <p>Calibration mixer/filter combination (must be reciprocal $S_{21} = S_{12}$.)</p>
	<p>Common equipment for both SMC and VMC</p> <ul style="list-style-type: none"> • Mechanical cal kit or ECal module 	

[See Comparison of Mixer Characterization using New Vector Characterization Techniques.](#)

Requirements and Limitations

The following PNA-X features are **NOT** available with FCA:

- Analog Sweep ([Stepped sweep](#) mode only)

- [Log frequency](#) sweeps
- [ECal User Characterization](#) (can NOT be created in FCA channel)
- [Time Domain](#)
- [Balanced measurements](#)
- [Interface Control](#)
- [Port extensions](#)
- [Some Fixturing Features](#)
- [External Test Set Control](#) (Option 551)
- [PMAR \(Power Meter As Receiver\)](#)

How to make SMC or VMC Measurements

The following is an overview of how to make an FCA measurement:

1. DECIDE to use either a SMC or VMC measurement. [See a comparison of these two measurement types.](#)
2. CREATE an SMC or VMC Measurement.
3. [SETUP](#) the measurements.
4. CALIBRATE your SMC or VMC measurement.

Create an SMC or VMC Measurement

1. On the PNA-X front panel, press **Meas** then **[Measurement Class]**
2. Select **SMC or VMC**, then either:
 - **OK** delete the existing measurement, or
 - **New Channel** to create the measurement in a new channel.
3. The default SMC or VMC measurement is displayed.
4. See [SMC measurements](#) or [VMC measurements](#) to learn about the parameters that are offered in each.

How to make SMC or VMC settings

To provide quicker access, use the Setup softkey. [Learn how.](#)

Using front-panel HARDKEY [softkey] buttons

1. Press **FREQ**
2. then **[Mixer Setup]**

Using a mouse with PNA Menus

1. Click **Stimulus**
2. then **Freq**
3. then **Mixer Setup**

Programming Commands

Valid Mixer Configuration / Sweep Type Combinations

Configuring the SMC and VMC Setup dialog can be challenging at first. **RED** messages like this one appear at the bottom of the Setup dialog to notify you of an invalid setup.

Unsupported mixer configuration and sweep type

At least one range (Input, LO, or Output) **MUST** be Fixed.

The following are the **Valid Mixer Configurations**:

Sweep Type	Input	LO	Output
Linear	Swept	Swept	Fixed
	Swept	Fixed	Swept
	Fixed	Swept	Swept
CW Time	Fixed	Fixed	Fixed
Power			

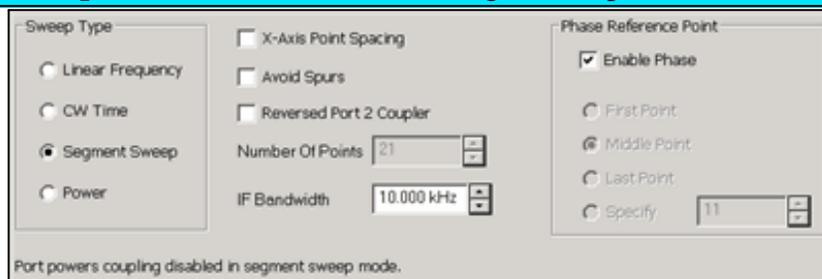
Tips

Although you will soon become comfortable navigating these tabs, at first it may be best to complete the dialog in the following order:

1. For 2-stage mixers, select [Mixer Setup](#) settings.
2. Select [Sweep tab](#) settings.
3. Select [Mixer Frequency](#) settings.
4. Select [Power](#) settings.
5. Select [Mixer \(LO\) Power](#) settings.

The following FCA settings are common to VMC and SMC:

Sweep Tab - SMC and VMC dialog box help



Sweep Type

Linear Sweep frequency. Measurements are displayed on a standard grid with ten equal horizontal divisions. Learn how to [select the range to display on the X-Axis](#).

CW Time All ranges are set to a Fixed (CW) frequency, and the data is displayed versus time.

Segment Sweep Sweep user-defined segments. [Learn more](#).

Power Sweep Input or LO power.

X-axis Point Spacing (Available only with Segment Sweep) - [Learn about this feature](#)

Avoid Spurs - [Learn about this feature](#).

Reversed Port 2 Coupler (SMC ONLY) Check when making SMC measurements with the Port 2 coupler reversed. [Learn how](#). Do this to increase power at low frequencies. Checking this box has the following effects:

- The measurement is faster because Reduce IFBW is no longer performed on SC21 measurements.
- The SMC Calibration no longer attempts to level the Port 2 power (it is not possible to level the power through the coupler at those low frequencies). This keeps the PNA from producing “source unlevelled” errors on Port 2.

Number of Points [Learn about this feature](#).

IF Bandwidth [Learn about this feature](#).

Phase Reference Point (SMC ONLY) [Learn about this feature](#).

Save...

Load...

[Learn about these buttons.](#)

Power Tab - SMC and VMC dialog box help

Note: Set LO Power on the [Mixer \(LO\) Power tab](#).

Power On (All Channels) Port Powers Coupled

DUT Input Port
Input Port: Port 1
Power Level: -15.00 dBm
Source Attenuator: Auto 5 dB
Receiver Attenuator(A): 0 dB
Source Leveling Mode: Internal

DUT Output Port
Output Port: Port 2
Power Level: -15.00 dBm
Source Attenuator: Auto 5 dB
Receiver Attenuator(B): 0 dB
Source Leveling Mode: Internal

DUT Input Port Power Sweep
Start Power: -15.00 dBm
Stop Power: -15.00 dBm
Points: 201
Power Step: 0.000 dB

DUT Output Port Power Sweep
Start Power: -15.00 dBm
Stop Power: -15.00 dBm
Points: 201
Power Step: 0.000 dB

Path Configuration...

Configures Input and Output power settings for an FCA measurement. Use the [Mixer Power tab](#) to set LO power.

Power ON (All channels) Check to turn RF Power ON or clear to turn power OFF for all channels.

Port Powers Coupled Check to set the same power level at the DUT Input and Output ports. The LO power is NOT coupled. Clear to set power levels independently for each test port. Uncouple power, for example, to apply more power in the reverse direction than in the forward direction
Learn more about [Setting Independent Port Power](#)

DUT Input / Output Port

Select the PNA port that is connected to the DUT Input and Output. For **VMC**, the DUT input must always be connected to PNA port 1 because of the need for a [reference mixer](#) on port 1.

Power Level Set the power level to the DUT Input port. To set power at the Output port, clear the **Port Powers Coupled** checkbox.

Source Attenuator Auto Check to automatically select the correct attenuation to achieve the specified input power. Clear, then select attenuator setting that is used achieve the specified Power Level. [Learn more about Source Attenuation.](#)

All PNA channels in continuous sweep must have the same attenuation value. [Learn more.](#)

Receiver Attenuator Specifies the receiver attenuator setting for the DUT port.

Source Leveling Choose from: **Internal** (normal operation), **Open Loop** (used only for [Wideband Pulse measurements](#)), or **Receiver - R1** for [Receiver Leveling](#).

DUT Input and Output Port Power Sweep

Available when Power (sweep) is selected on the [Sweep tab](#).

Input Start and Stop Power To set Start and Stop power at the Output port, clear the **Port Powers Coupled** checkbox.

Note: If your DUT requires more input power than this setting allows below 3.2 GHz, use the PNA-X **Hi-power mode**, available from the [RF Path Configuration](#) dialog. The disadvantage to this is higher harmonic content.

Power Points Number of power points to measure.

Power Step (Size) Calculated value from current Start, Stop, and Points settings. This setting can NOT be changed directly.

Path Configuration click to launch the [RF Path Configuration](#) dialog.

Save...

Load...

[Learn about these buttons.](#)

The following tabs are shared with all [Mixer / Converter Applications](#):

- [Mixer Frequency tab](#)
- [Mixer \(LO\) Power tab](#)
- [Mixer Setup tab](#)

Mixer Frequency tab - SMC and VMC Setup -dialog box help

Mixer Frequency

Input	Start/Stop	1.00000000 GHz	2.00000000 GHz	Calc Input
LO1	Fixed	500.000000 MHz	<input checked="" type="checkbox"/> Input > LO	
IF	Start/Stop	<input checked="" type="radio"/> + 1.50000000 GHz	2.50000000 GHz	Calc LO1
		<input type="radio"/> - 500.000000 MHz	1.50000000 GHz	Calc LO2
LO2	Fixed	500.000000 MHz	<input checked="" type="checkbox"/> IF1 > LO2	
Output	Start/Stop	<input checked="" type="radio"/> + 2.00000000 GHz	3.00000000 GHz	Calc Output
		<input type="radio"/> - 1.00000000 GHz	2.00000000 GHz	

[Learn about this dialog](#)

Mixer Setup tab - SMC and VMC Setup -dialog box help

Converter Stages: 2

Hardware Configuration
Port 3: Thru
Port 4: Thru

Converter Model: Single Stage

Port 1

Port 2

LO1: Not controlled

LO2: Not controlled

[Learn about this dialog](#)

Mixer (LO) Power tab - SMC and VMC Setup -dialog box help

Power On (All Channels)

LO1: Not Controlled
LO1 Power: -10.00 dBm
Source Leveling Mode: [Dropdown]

LO2: Not Controlled
LO2 Power: -10.00 dBm
Source Leveling Mode: [Dropdown]

Port Settings

Port 3
Source Attenuator: 5 dB
Receiver Attenuator: 0 dB

Port 4
Source Attenuator: 5 dB
Receiver Attenuator: 0 dB

Swept Power Settings

	Start	Stop	Step
LO1 Swept Power:	-10.00 dBm	0.00 dBm	0.050 dB
LO2 Swept Power:	-10.00 dBm	0.00 dBm	0.050 dB

Path Configuration...

[Learn about this dialog](#)

FCA Segment Sweep

The following settings appear on the Mixer Frequency tab when **Segment Sweep** is selected on the [Sweep tab](#).

Mixer Frequency tab - Segment Sweep - SMC and VMC dialog box help

Mixer Frequency

Segment: 1 of 3 State: ON Add Delete

Input: Start/Stop 1.000000000 GHz 2.000000000 GHz Calc Input

LO1: Fixed 500.000000 MHz Input > LO

Output: Start/Stop + 1.500000000 GHz 2.500000000 GHz Calc LO
- 500.000000 MHz 1.500000000 GHz Calc Output

Num Pts: 21 Input Port: -10.00 dBm LO1 Port: -10.00 dBm

IF Bandwidth: 100.000 kHz Output Port: -10.00 dBm

How to configure a segment:

1. Click **Add**. Click **Delete** to remove a segment and renumber all subsequent segments.
2. State is **ON** by default. Click **OFF** and that segment will not be included in the sweep.
3. Configure **Frequency settings** for Input, LO, and Output ranges.

- For each segment, the same [sweep requirements](#) apply as a standard (non-segment) sweep. For example, at least one range MUST be Fixed (Start = Stop frequencies).
- The Input, Output, and LO frequencies of segments ARE allowed to overlap other segments.
- All segments must sweep in either the forward (Start<Stop) or reverse (Start>Stop) directions. Mixed sweep directions are NOT allowed.
- The following settings can be set independently for each segment:
 - [Number of Points](#) - Total number of points for all segments is limited to the [Max allowed by the PNA](#).
 - [IF Bandwidth](#)
 - **Port Powers:** (Input, Output, LO 1, LO2). These settings override the settings on the [Power tab](#).
- The following settings apply to ALL segments:
 - Number of Converter/Mixer Stages (1 or 2).
 - LO Source Selections
 - [All Input and LO Multipliers and Dividers](#)
 - [Source and Receiver Attenuator Settings](#)
 - [Source Leveling](#)
 - [Avoid Spurs](#)
 - [Nominal Incident Power](#) (SMC only)
 - [X-Axis Display](#) (Input, LO1, LO2, Output) There must be at least two data points for this setting to be available.
 - [X-Axis Point Spacing](#) (vs Normal point spacing).
 - [SMC + Phase](#)
- Mixer Segment sweep data can be saved to a ***.S2PX** file (NOT *.S2P).
- Mixer Segment setup information is saved to a ***.MXRX** file. [Learn more](#).

Save...

Load...

[Learn about these buttons](#)

Apply and Interpolate FCA Cal Sets

In general, when a Cal Set covers a wider frequency range than the channel, the PNA will offer to

interpolate the Cal Set when it is applied. [Learn more](#). However, with FCA measurements the LO frequency range may also be considered.

- [VMC measurements](#) ALWAYS CONSIDER the LO frequency range and performs interpolation if possible. If the LO frequency range of the measurement is NOT within the LO frequency range of the Cal Set, then the Cal Set can NOT be applied.
- [SMC measurements](#) ALWAYS IGNORE the LO frequency range. Therefore, if the Input and Output frequency ranges of the measurements are within those of the Cal Set, then the Cal Set is interpolated if necessary and applied. For example, this would allow you to perform ONE SMC calibration with Input range = the PNA frequency span, LO at 0 Hz, and Output range + the PNA frequency span. This Cal Set could be applied to ALL SMC measurements. [Learn more about applying SMC al Sets](#).

These same general concepts apply to [segment sweeps](#). However, if ALL applicable frequency ranges (SMC: Input and Output and VMC: Input, Output, and LO) are NOT within those ranges of the measurement for ONE segment, then the Cal Set is NOT applied for ANY segment.

Select X-axis Display for FCA Measurements

Click **Response**, then **Measure**, then **Select X-axis**, then **Input, LO**, or **Output**

When **Sweep Type = Linear**, you can choose to show the frequency range of any of the swept parameters on the X-axis.

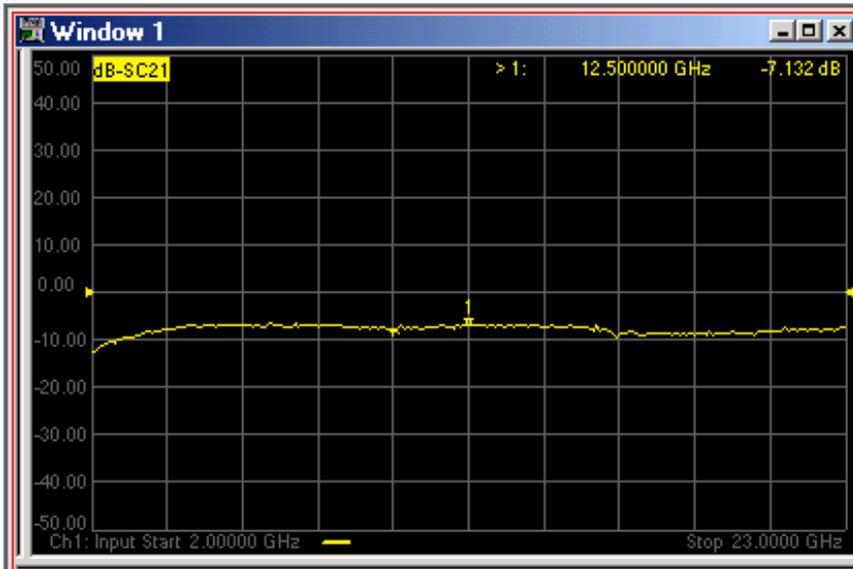
For example, the following image shows an SMC Fixed Output response with the **Input frequency range** on the X-axis:

Output: 100 MHz (data trace)

Input: 2 GHz to 23 GHz (X-axis)

LO: 1.9 GHz to 22.9 GHz (not shown)

Marker annotation shows Output power at Input frequency.



Save Trace Data

You can save your FCA measurement data in several standard formats.

Click **File**, then **Save Data As** to save your FCA data.

The following shows how CSV and SNP files are saved.

Mixer Trace Data

When you select **Mixer Trace Data**, the FCA data is saved to a CSV file in the following format:

```
#MIXER TRACE FILE,A.01.00
```

```
SegIndex, InputFreq, OutputFreq, LO1Freq, InputPower, LO1Power, SC21  
Mag (dB), SC21 Phase (Deg)
```

SNP Format

Each record contains 1 stimulus value and 4 parameters (total of 9 values) as follows:

```
Stim Real(p1) Imag(p1) Real(p2) Imag(p2) Real(p3) Imag(p3) Real(p4) Imag(p4)
```

where **pX** is the parameter depending on measurement type:

Measurement Type	p1	p2	p3	p4
Scalar	S11	SC21 (FWD)	SC12 (REV)	S22
Vector	S11	VC21	VC12	S22
Mixer Characterization	Directivity	Source Match	Reflection Tracking	M21

- If correction is OFF, data is only saved for the active parameter. Zeros are saved for all other parameters.
- If correction is ON, data is saved for all of the parameters.

All files contain the following Header Information: Brackets [] contain parameters.

```
!Agilent [Instrument Model Number]: [version]
!Mixer S2P File: [Mixer Measurement Type]
!Parameters: [Parameter List]
!Calibration State: [On/Off]
```

```
!# Begin Mixer Setup
![Mixer Setup parameters listed here]
![Mixer Parameter 1]
.
.
![Mixer Parameter n]
!# End Mixer Setup
```

```
# [S2P data here]
```

Avoid Spurs

On the [Mixer Setup](#) dialog box, check **Avoid Spurs**

The Avoid Spurs feature of the Frequency Converter Application attempts to prevent unwanted mixing products from appearing on the PNA screen. The Avoid Spurs feature does not significantly impact measurement speed.

Note: The Avoid Spurs feature is OFF by default for FCA calibrations. For highest accuracy, make measurements with the Avoid Spurs feature at the same state (ON or OFF) as was used when calibrating.

Description

A spur, or spurious signal, is a term used to describe the unwanted product of two signals mixing

together. When you configure the mixer setup dialog box for a desired Output, the PNA computes the frequencies of potential unwanted signals. By manipulating internal PNA hardware, these signals are avoided and do not appear on the PNA display. This means you do not need to use external filters to prevent spurious signals from appearing on the PNA display.

The time required for the PNA to compute the frequencies of unwanted spurious signals MAY be noticeable depending on the number of data points in your measurement. However, once computed, the time required for the PNA to avoid the spurs is usually insignificant.

Limitations

The Avoid Spurs utility cannot avoid every spur. However, when there is a choice of spurs to avoid, it will avoid the largest spur.

The Computation of Avoided Spurs

The Avoid Spur computer avoids the following spurs:

- LO, and its interaction with internal PNA components, and 16 of its harmonics.
- Input frequencies and 16 of its harmonics.
- Undesired Image frequencies (Sum or Difference) and 16 of its harmonics.

Last modified:

11-Mar-2013	Added Reverse Coupler setting
13-Dec-2011	Removed unratioed and modified User Char in limitations list
18-Oct-2011	Added PMAR limitation
27-Apr-2011	Removed Copy Channels limitation
9-Dec-2010	Updated for A.09.33
26-Aug-2010	Add link to Cal 2-stage SMC+Phase
25-Mar-2010	Add SMC with phase, Data Save As, GAV
5-Feb-2010	Add Speed SMC
3-Sep-2008	Removed legacy content
1-May-2008	Updated for selectable ports
5-Oct-2007	Added link to embedded LO

Scalar Mixer/Converter Measurements (SMC)

SMC Setup and Calibration is very similar to [VMC](#). See [FCA Overview](#) to learn about the features that are common to these two applications.

The following information is unique to SMC:

- [SMC Hardware Setup](#)
- [Create an SMC Measurement](#)
- [SMC Parameters Offered](#)
- [The SMC Mixer Setup dialog](#)
- [Speed Up SMC Measurements](#)
 - [Use Nominal Incident Power](#)
 - [Apply a Cal Set or SMC Cal Type](#)
 - [Reverse Port 2 Coupler below 55 MHz](#) (separate topic)
- [SMC Calibration](#)
- [SMC + Phase](#) (separate topic)

See Also

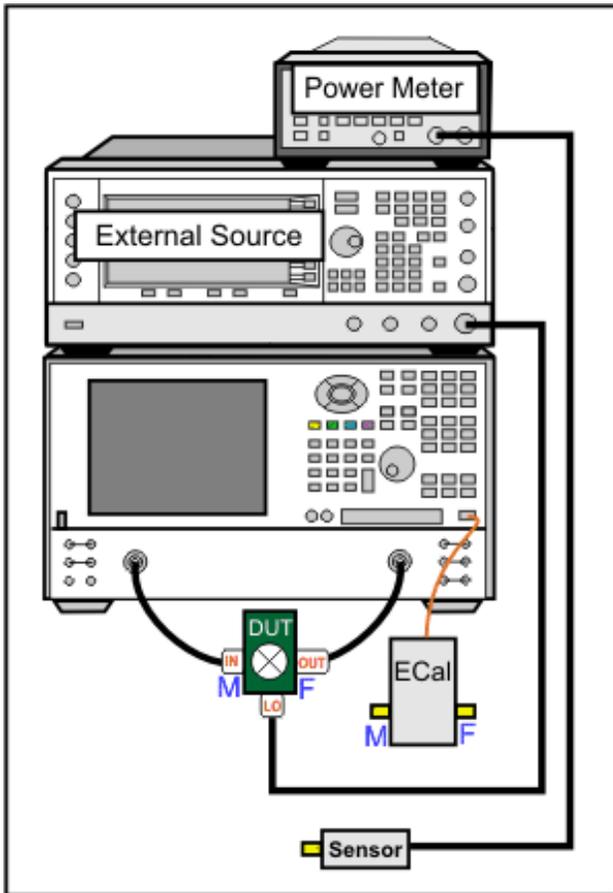
[Embedded LO](#)

[How to make an SMC Fixed Output Measurement](#)

[SMC with a Booster Amp](#)

[Programming Commands](#)

SMC Hardware Setup



SMC requires a power meter/sensor, two sources, and a Cal Kit or ECal module

- Your DUT can be connected to any PNA ports. [Learn more.](#)
- When using PNA-X with Internal Second Source, the external source is NOT necessary. [Learn which PNA ports can be used for the LO.](#)
- Connect **External Source** to the PNA GPIB Controller port. Learn how to [Configure an External Source.](#)
- Connect the [10 MHz reference signal](#) of an external source to the PNA. This is especially important with SMC + Phase measurements.

Use either a GPIB power meter or USB power sensor.

How to configure two power sensors to cover the SMC measurement frequency range.

Using a dual channel power meter, with both sensors connected:

1. At the SMC [Select DUT Connectors](#) dialog, click **Source Cal Settings**
2. At [Source Calibration Settings](#) dialog, click **Power Meter Config**
3. At [Power Meter Settings](#) dialog, click **Sensors**
4. At [Power Sensor Settings](#) dialog, clear the "Use this sensor only..." checkbox for both sensors.
5. Then enter the **Min** and **Max Frequencies** for both sensors.

During the SMC Cal, you will be prompted to connect each sensor at the appropriate time.

Create an SMC Measurement

1. On the PNA-X front panel, press **Meas** then **[Measurement Class]**
2. Select **SMC**, then either:
 - **OK** delete the existing measurement, or
 - **New Channel** to create the measurement in a new channel.
3. An SC21 measurement is displayed.

SMC Parameters Offered

To select additional parameters to display, click **Trace**, then **New Trace**, then select a parameter from the list.

Important Note: Connecting your DUT to the PNA:

RF and **IF** terminology is NOT used in FCA because the PNA does not know how the DUT is labeled or how it will be used. Instead, the general terms INPUT and OUTPUT are used.

- **INPUT** - The DUT port being stimulated with frequencies before conversion.
- **OUTPUT** - The DUT port outputting converted frequencies.

INPUT and OUTPUT Frequencies are specified using the [Mixer Setup dialog box](#).

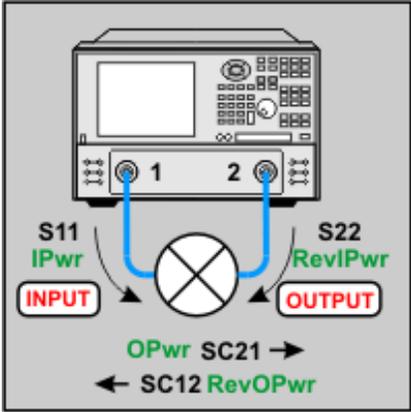
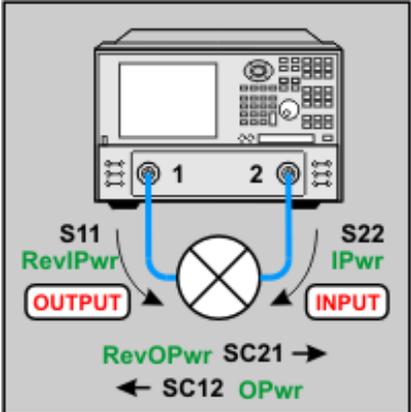
The DUT input and output can be connected to any PNA ports.

Note: Although there are MANY configuration possibilities, the following images and descriptions show ONLY a DUT connected to PNA ports 1 and 2.

Legend:

Black are ratioed measurements (test port/reference receiver).

Green are unratioed measurements (either a test port OR reference receiver).

<p>DUT Input to PNA port 1</p> <p>DUT Output to PNA port 2</p>	<p>DUT Input to PNA port 2</p> <p>DUT Output to PNA port 1</p>
	
<p>Ratioed</p> <ul style="list-style-type: none"> • SC21 (Conversion Loss) Stimulus at Input, response at Output (B/R1). • SC12 (Reverse Isolation) Stimulus at Output, response at Input (A/R2) 	<p>Ratioed</p> <ul style="list-style-type: none"> • SC12 (Conversion Loss) Stimulus at Input, response at Output (A/R2) • SC21 (Reverse Isolation) Stimulus at Output, response at Input (B/R1)

<ul style="list-style-type: none"> • S11 (Input match) Stimulus and response at Input (A/R1) • S22 (Output match) Stimulus and response at Output (B/R2) 	<ul style="list-style-type: none"> • S11 (Output match) Stimulus and response at Output (A/R1) • S22 (Input match) Stimulus and response at Input (B/R2)
<p>Unratioed Absolute test port receiver measurements. The receiver is automatically selected depending on the DUT configuration.</p> <ul style="list-style-type: none"> • IPwr (Incident Power) - stimulus and response at Input. • RevIPwr (Reverse Incident Power) - stimulus and response at Output. • OPwr (Output Power) - stimulus at Input, response at Output. • RevOPwr (Reverse Output Power) - stimulus at Output, response at Input. 	

SMC Mixer Setup

How to start the SMC Mixer Setup dialog

To provide quicker access, use the Setup softkey. [Learn how.](#)

Using front-panel HARDKEY [softkey] buttons

1. Press **FREQ**
2. then **[Mixer Setup]**

Using a mouse with PNA Menus

1. Click **Stimulus**
2. then **Freq**
3. then **Mixer Setup**

Programming Commands

The following SMC Mixer Setup dialog tabs are presented:

- [Sweep Tab](#) (shared with VMC)
- [Power Tab](#) (shared with VMC)
- [Mixer Freq Tab](#) (shared with all converter apps)
- [Mixer Power Tab](#) (shared with all converter apps)
- [Mixer Setup Tab](#) (shared with all converter apps)

Speed Up SMC Measurements

Using default SMC settings, any calibrated SMC measurement requires four sweeps. However, you can

reduce the number of sweeps required by selecting one or more of the following settings.

- [Use Nominal Incident Power](#)
- [Apply Cal Set or Cal Type](#)
- To speed up a Swept LO measurement when using an external source for the LO, use **Hardware List (BNC)** Trigger setting . [Learn more.](#)
- [Reverse Port 2 Coupler below 55 MHz](#) (separate topic)

Use Nominal Incident Power

Click **Response**, then **Measure**, then **Use Nominal Incident Power**

Each data sweep of a fully corrected SMC transmission measurement actually requires FOUR data sweeps. When you clear **Use Nominal Incident Power**, the reference receiver (R1 or R2) does NOT measure incident power. Instead, the incident power is assumed to be at the level that was set with the [Source Power Calibration](#) that is done as part of every SMC measurement. The degradation in accuracy is very negligible if the input or output of your DUT is well-matched.

This selection eliminates sweeps ONLY when both **Include Input Match** AND **Include Output Match** is cleared on the Cal Type dialog. [Learn more.](#)

Apply a Cal Set or SMC Cal Type

You can create an FCA measurement and apply an existing Cal Set as you can with any PNA measurement. Learn about [Cal Sets](#). In addition, from a Cal Set, you can apply a specific SMC Cal Type to an existing SMC measurement.

How to apply an SMC Cal Type

1. [Create an SMC measurement](#)
2. Calibrate or apply an existing SMC Cal Set, then...

Using front-panel HARDKEY [softkey] buttons

1. Press **CAL**
2. then **[Manage Cals]**
3. then **[Correction Methods]**

Using a mouse with PNA Menus

1. Click **Response**
2. then **Cal**
3. then **Manage Cals**
4. then **Correction Methods**

[Programming Commands](#)

Correction Method dialog box help



By default, each SMC calibration requires FOUR sweeps. Clearing boxes will eliminate sweeps and speed up your SMC measurements. The difference in speed is most noticeable when making fixed input or fixed output measurements with an external LO source.

Include input match correction Check to perform a sweep to measure and correct for INPUT match. Clear this box if the input of your mixer is well-matched to the PNA, or if your setup does not permit a valid S11 measurement.

Include output match correction Check to perform a sweep to measure and correct for OUTPUT match. Clear this box if the output of your mixer is well-matched to the PNA, or if your setup does not permit a valid S22 measurement.

Include SC12 Sweep Check to perform a reverse sweep to measure SC12.

- When checked (default setting), a calibrated SMC measurement sweeps in both forward (SC21) and reverse (SC12) directions.
- Clear this checkbox to eliminate sweeps in the reverse direction. This means that the following measurements will NOT be corrected: SC12, RevOPwr, RevIPwr.

Corrected Parameters Lists the parameters that **can** be corrected given the boxes that are currently checked. These parameters may not be currently measured.

Calibration Type Shows the type of SMC Cal that will be applied given the boxes that are currently checked.

Learn about [Use Nominal Incident Power](#)

How many sweeps can be eliminated?

Setting	Parameters Learn about parameter abbreviations	# of sweeps
ALL checked and clear Use Nominal Incident Power	IPwr,OPwr,RevIPwr,RevOPwr,SC21,SC12,S11,S224	Total
Perform this action...	to REMOVE these parameters...	and these sweeps
Clear "Include SC12"	Remove RevIPwr,RevOPwr,SC12	Removes 1
Clear "Include OUTPUT match"	Remove S22	Removes 1 when Nominal is checked*.
Clear "Include INPUT match"	Remove S11	Removes 1 when Nominal is checked*.
Check "Use Nominal Incident Power"	Remove IPwr, RevIPwr	May remove up to 2*
ALL cleared and check Nominal Incident Power	OPwr,SC21	1 Total

*S11 shares a sweep with IPwr and S22 shares a sweep with RevIPwr. Therefore, when **Include Input Match** or **Include Output Match** is checked, then checking Nominal incident power does nothing.

[VMC measurement](#) sweeps can NOT be eliminated.

SMC Calibration Overview

The [SMC Calibration Wizard](#) guides you through this process.

When applying a [Phase Reference cal set](#), step 1 (power cal) is NOT performed.

1. Connect a power meter / sensor to PNA Port 1. At each step of the input and output frequency, the PNA measures:
 - input match of the power sensor
 - source power of the PNA
2. Perform two Full 2-port calibrations: one over the INPUT frequencies and one over the OUTPUT frequencies of the DUT. (If your DUT is a linear device, the calibration uses only the INPUT frequency range.) Use either a mechanical calibration kit or an ECal module.

For Mixers / Converters with High-output Power

The Unknown Thru method is NOT valid when there is over 40 dB of combined loss in the Unknown Thru and calibration path. In this case, the following calibration and correction method is recommended.

- On the Cal Wizard [Modify Frequency](#) page, select Defined Thru or Flush Thru as the [Thru method](#). When using an ECal module, also on the Modify Frequency page, disable (clear) **Do Orientation** due to very low power.
- After calibration, on the [Correction Method](#) dialog, CLEAR the **Include output match correction** and **Include SC12 Sweep** check boxes. Check ONLY **Include input match correction**.
- To learn more about High-power measurements, see our [App Notes](#).

SMC Cal Wizard

The following dialog boxes are presented during an SMC Calibration.

Indented steps are optional.

- [Calibration Setup](#)
 - [Waveguide/In-fixture/On-Wafer Setup](#)
- [Select DUT Connectors and Cal Kits](#)
 - [Modify Frequency Cal](#)
 - [Specify how the ECal module is connected](#)
- [Power Cal Settings](#)
- [SMC Cal Steps](#)
- [Calibration Completed](#)
- [Specify Adapter Delay](#)

How to Perform a SMC Calibration

1. [Create an SMC measurement](#), then...

Using front-panel HARDKEY [softkey] buttons

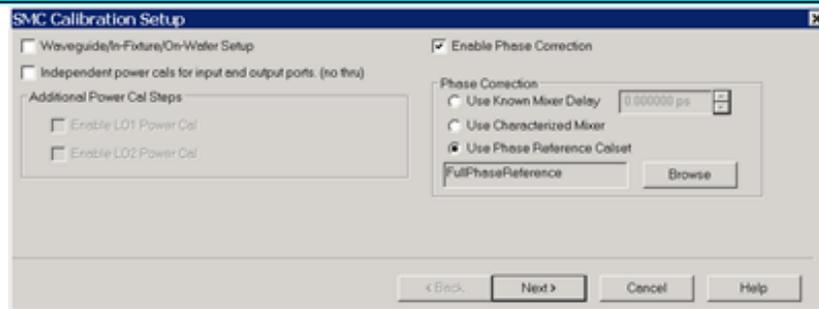
1. Press **CAL**
2. then [**Start Cal**]
3. then [**Cal Wizard**]

Using a mouse with PNA Menus

1. Click **Response**
2. then **Cal Wizard**

Programming Commands

SMC Calibration Setup dialog box help



Allows you to review and change the settings for your SMC calibration.

Note: With release A.09.90 and before, checking both '**Independent power calcs**' AND '**Use Phase Reference Calset**' would generate an error after performing the calibration. With releases AFTER A.09.90, the two settings are compatible.

Waveguide/In-fixture/On-Wafer Setup Click **Next** to launch the following Setup dialog box.

Independent power calcs for input and output ports (no thru) Check if a Thru standard is NOT available. During the power cal, you will be prompted to connect the power sensor to the Input, then the Output port.

Additional Power Cal Steps

Enable LO1 / LO2 Power Cal Check when LO1 / LO2 is controlled (on the [Mixer Setup](#) tab) to perform a Power Cal on the LO source(s).

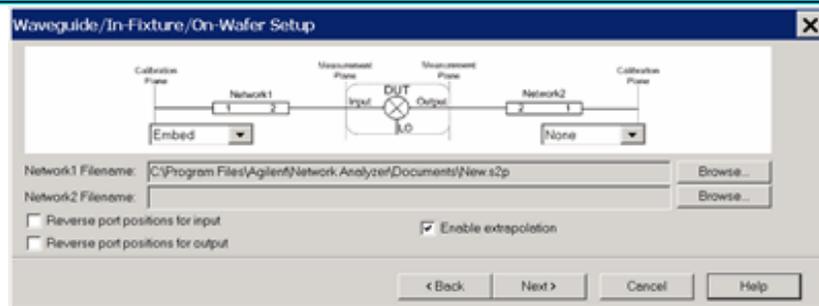
Phase Correction

Enable Phase Correction Check to enable Phase measurements.

Choose one of the following methods to specify the delay through the characterized mixer. With the first two methods, the phase delay through a Calibration Mixer is measured and compared to the known delay, either entered, or stored in an *.S2P file.

- **Use Known Mixer Delay** Enter the fixed, known, delay through the calibration mixer.
- **Use characterized mixer** Select, then browse to the *.S2P file that characterizes the calibration mixer. Use an *.S2P file when making segmented SMC + Phase measurements. [Learn more](#). Use either of the following two methods to characterize the Cal Mixer over the SMC measurement frequency range:
 1. Use the [Mixer Characterization Wizard](#). (Click **Response**, then **Cal**, then **Mixer Characterization Wizard**.) The Cal Mixer has the same requirements as the **VMC Cal Mixer**. [Learn more](#).
 2. In a calibrated VMC channel, measure the group delay of the calibration mixer, then save to an *.S2P or *.S2PX file. However, a characterized mixer is required to calibrate the VMC channel.
- **Use Phase Reference Calset** Select, then browse to the Phase Reference Calset that covers the frequency range of the current measurement. [Learn more about SMC with a Phase Reference Calibration](#).

Waveguide/In-fixture/On-Wafer Setup dialog box help



This dialog box appears **ONLY** if you checked the **Waveguide/In-fixture/On-Wafer Setup** box in the previous [Cal Setup](#) dialog.

Allows you to embed or de-embed circuit networks on the input and output of your mixer under test.

For Network1 (Input) and Network2 (Output) select **Embed**, **De-embed**, or **None**.

Browse Click to navigate to the .S2P file that models the network to embed or de-embed.

Reverse port positions for input/output Check to cause the Fixture/Adapter to be configured with Port 2 connected to the PNA and Port 1 to be connected to the DUT. The image in the dialog is updated to reflect that change.

Enable Extrapolation Check (default setting) to apply a simple extrapolation when the S2P file has a narrower frequency range than the channel. The values for the first and last data points are extended in either direction to cover the frequency range of the measurement. A warning message is also displayed when extrapolation is necessary.

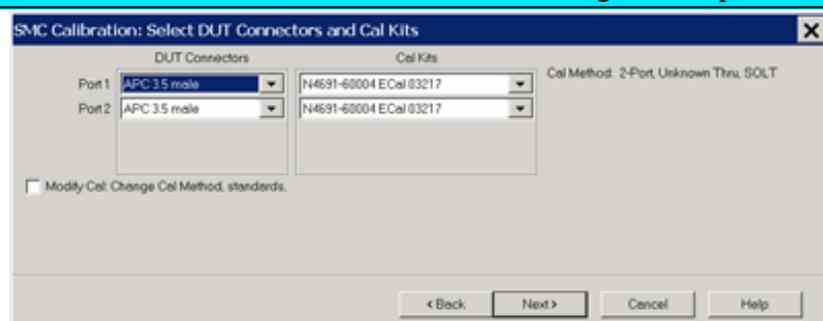
To Embed or De-embed

- When you have a 2 port network that needs to be connected between the Cal reference plane and the DUT during the measurement, but it is NOT present during the calibration, then that network has to be **De-Embedded** from the port in question during the calibration. In other words, De-Embedding in FCA calibration extends the calibration reference plane to include the two port network.
- When you have a 2 port network that is included as part of the calibration reference plane but has to be disconnected during the measurement, then that 2-port network has to be **Embedded** for the port in question during the calibration. In other words, Embedding in FCA calibration retracts the calibration reference plane to exclude the two port network during the measurement.

Notes

- [Characterize Adaptor Macro](#) can be used to create the S2P file.
- Interpolation is performed when more frequencies are included in the file than in the channel, and the data points do not exactly match those of the measurement.

Select DUT Connectors and Cal Kits dialog box help



Allows you to specify the connector type and Cal Kit for each DUT port.

Port n For each listed PNA port, specify the DUT connector type and gender, and the Cal Kit to use.

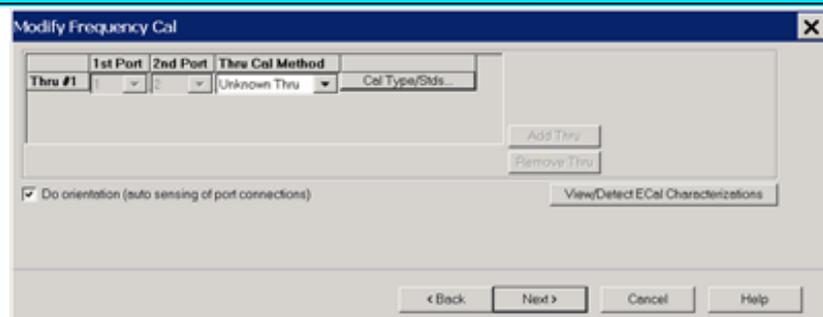
Note: If your DUT connectors are:

- **Waveguide** Change the system impedance to 1 ohm before performing a calibration. See [Setting System Impedance](#).
- **Not listed** (male and female) Select **Type A** as the connector type. Type A requires a calibration kit file containing the electrical properties of the standards used for calibration (see [Calibration kits](#)).
- **Unspecified** (like a packaged device) Select **Type B** as the connector type. Type B requires a calibration kit file containing the electrical properties of the standards used for calibration (see [Calibration kits](#)).

Modify Cal Check, then click **Next**, to start the [Modify Frequency Cal dialog](#).

Source Cal Settings Click to start the [Source Cal Settings](#) dialog.

Modify Frequency Cal dialog box help



This dialog appears only when **Modify Cal** is checked on the previous dialog.

Thru Cal Method For each Thru connection, choose the Thru method. [Learn more about these choices](#).

Cal Type/Std Click to start the [Modify Calibration Selections](#) dialog box.

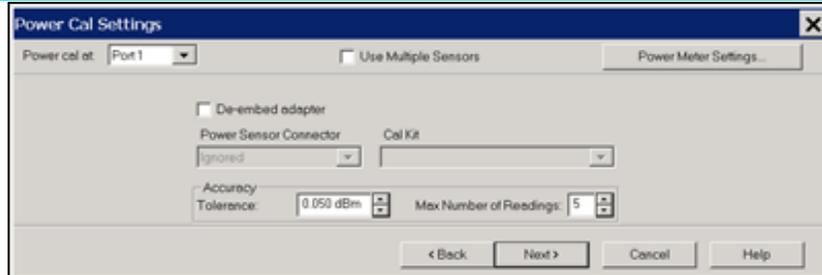
The following selections are available ONLY if using an ECal module.

Do orientation When this box is checked (default) the PNA senses the ECal model and direction in which the ECal module port is connected to the PNA ports. If power to the ECal module is too low, it will appear as if there is no ECal module connected. If you use low power and are having this problem, clear this check box to provide the orientation manually. Orientation occurs first at the middle of the frequency range that you are calibrating. If a signal is not detected, it tries again at the lowest frequency in the range.

[See note](#) about calibrating for high-output mixer/converters using an ECal module.

View/Detect ECal Characterizations Appears only if an ECal module is selected for use. Click to invoke the [View ECal Modules and Characterizations](#) dialog box. Displays a list of ECal modules that are connected to the PNA.

Power Cal Settings dialog box help



Note: A **Use Power Table** checkbox (not shown) is available when a mmWave SMC measurement is active. [Learn more.](#)

Power Cal at: Select the source port for which a Power Calibration will be performed. The source and receiver correction will be transferred to all other sources and receivers involved in the S-parameter measurements.

Use Multiple Sensors (NOT available with mmWave SMC measurements.) Check this box when you want to use more than ONE power sensor to cover the measurement frequency range. The dialog is replaced with the [Multiple Sensors](#) dialog (see following image). When "Use Multiple Sensors" is cleared (default setting), connect only ONE sensor to the PNA.

Power Meter Settings Click to start the standard [Power Meter Settings dialog](#).

De-embed (power sensor) adapter When the power sensor connector is NOT the same type and gender as the DUT connector for the specified port, then for optimum accuracy, extra cal steps are required to measure and correct for the adapter that is used to connect the power sensor to the reference plane.

Clear this box to NOT compensate for the added adapter.

Check this box to perform extra calibration steps to measure and correct for the adapter.

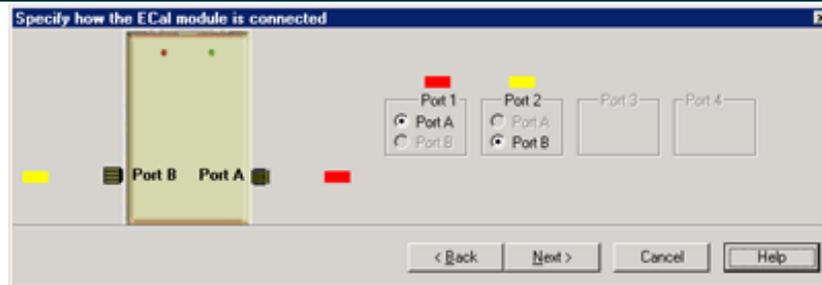
Then select the **Power Sensor Connector** type and gender of the power sensor. "Ignored" does NOT compensate for the added adapter, just as if the checkbox were cleared.

When this connector matches the DUT connector for the same port, then the PNA assumes that there is no adapter. Extra cal steps are NOT required and the Cal Kit selection is not available.

Otherwise, select the **Cal Kit** to be used to calibrate at the adapter.

See [Accuracy Settings](#) below.

Specify how the ECal module is connected dialog box help



This dialog box appears when the **Do orientation** checkbox in the previous **Modify Frequency** dialog box is cleared.

Click the ECal Port that is connected to each PNA port.

SMC Calibration Steps dialog box help



Power Level at which to perform the Power Cal.

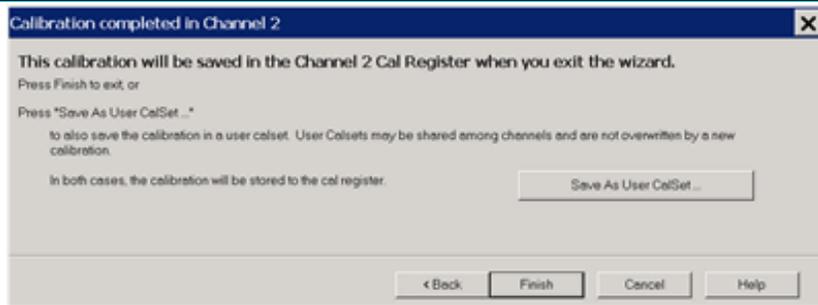
It is usually best to set power level to 0 dBm at the power sensor because the power sensor is calibrated at that level. Lower power levels will yield a slower and noisier calibration.

If an external component is used between the PNA-X test port and the calibration reference plane, then adjust the power level so that the power at the sensor is about 0 dBm if possible.

The current source attenuation value is shown on the dialog.

LO Power Cal (Optional) When [enabled](#), perform a Source Power Cal at the DUT LO connector. An LO must already be selected. [Learn how](#). The power level of the LO source calibration is set on the [\(LO\) Power Tab](#).

Calibration Completed dialog box help



Finish Save to the channel's calibration register.

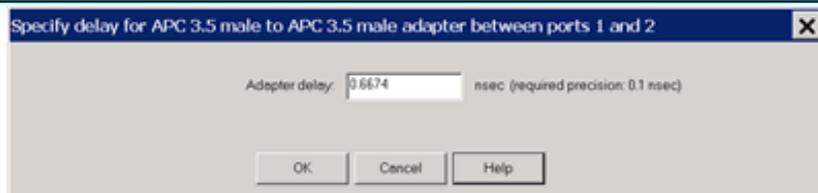
Save As User Cal Set Starts the [Save as User Cal Set dialog box](#) AND save to the channel's calibration register.

Cancel Calibration is NOT applied or saved.

Learn about [Calibration Registers](#).

Learn about [User Cal Sets](#)

Specify delay dialog box help



This dialog appears ONLY when [Adapter Removal](#) or [Unknown Thru](#) calibrations are performed.

The following values were estimated from the measurement. Most of the time, they are adequate. However, for CW sweep or frequency sweep with large step sizes, the accuracy of the values may be improved.

Adapter delay To improve this value, measure and record the delay of the adapter with a dense step size. Enter that value here. The required precision value is the accuracy that is required to characterize the delay value.

Nominal phase offset (Waveguide ONLY). To improve this value, measure and record the phase offset of the Waveguide adapter with dense step size. Enter that value here.

When one connector is coax and the other connector is waveguide, the phase offset has an ambiguity of 180 degrees. For consistency, the estimate provided here is always between 0 and 180 degrees. You can change this estimate to any value between -180 degrees and +180 degrees.

For SMC calibrations, this dialog box appears twice: once for the input frequencies and once for the output frequencies. The values can be slightly different.

Last modified:

29-Jan-2014	Added Calibration Setup pre 9.90 note
12-Mar-2013	Added Phase Ref image
14-Jan-2013	Added 10 MHZ reference signal connection
17-Oct-2012	Added high power note
6-Oct-2011	Clarify calibration mixer
8-Jun-2011	Clarify 'stable phase measurements'
1-Jun-2011	Add *.S2Px
25-Oct-2010	Combined SMC info to one topic and added fixturing features
26-Aug-2010	Add link to Cal 2-stage SMC+Phase
25-Mar-2010	Add SMC with phase, Data Save As, GAV
5-Feb-2010	Add Speed SMC
3-Sep-2008	Removed legacy content
1-May-2008	Updated for selectable ports
5-Oct-2007	Added link to embedded LO

SMC+Phase

Beginning with PNA Rev.A.09.20, you can optionally measure phase with SMC. This feature is available ONLY with Opt 083 on a PNA-X or N522x model. SMC + phase is NOT offered in the N523xA.

In this topic:

- [Overview](#)
- [How to make SMC+Phase measurements](#)
- [Measuring Mixers below 55 MHz](#)
- [Comparing SMC+Phase with VMC Phase Measurements](#)
- [How to improve the stability of SMC+Phase measurements](#)
- [How to Calibrate a 2-stage \(LO\) SMC+Phase Measurement](#)

See Also

[SMC Phase Reference Calibration](#)

[SMC Measurements and Calibration](#)

[SMC+Phase Demo](#) (Internet connection required)

Other FCA Topics

Overview

There are three methods used in the PNA to calibrate SMC+Phase measurements. All three methods rely on newer phase-coherent synthesizers in the PNA to produce phase capability in frequency offset measurements.

With the first two methods, during an SMC calibration, the phase delay through a Calibration Mixer is measured and compared to the known delay. The difference is used to correct subsequent SMC+Phase measurements.

A Calibration Mixer is required with the first two methods.

1. Enter the known delay into a dialog.
2. Uses the known delay at various frequencies from an *s2p file from a mixer characterization. You create the

*S2P file from a separate Mixer Characterization. This method is NOT supported with [Cal All](#).

3. Uses a Phase Reference to perform a 'tier 1' calibration. A Calibration Mixer is NOT required with this process. [Learn more about this process](#).

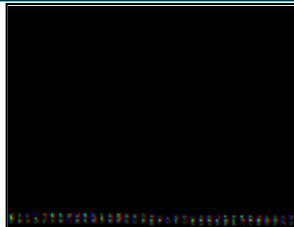
Notes

- A Reference Mixer is NOT required with any of these methods as it is with VMC.
- SMC+Phase can be measured on Converters with an Embedded LO. [Learn how](#).
- Phase can be measured with Power Sweeps.
- Phase can NOT be measured on Swept LO measurements.
- It is especially important with SMC + Phase measurements to connect the [10 MHz reference signal](#) of an external source to the PNA.
- With A09.90, you can now use a Phase Reference Cal Set with **Independent power cals for input and output ports (no thru)**. [Learn more](#).

How to make SMC+Phase measurements

1. On the Mixer [Sweep tab](#), check **Enable Phase**, then select the Phase Reference Point.

SMC Mixer Sweep tab - Phase Settings help



Enable Phase with SMC Check to perform phase measurements.

You can also enable phase measurements at the [SMC Cal Setup dialog](#).

Phase Reference Point

The SMC Phase measurement technique provides for a coherent phase relationship from one frequency to the next in each sweep. However, the phase measurement of the first data point is random from sweep to sweep. This initial phase offset does not impact measurements such as group delay or deviation from linear phase.

However, in order to keep a phase trace from appearing random, all phase data in the sweep is normalized against a single point. This results in a stable, normalized phase trace.

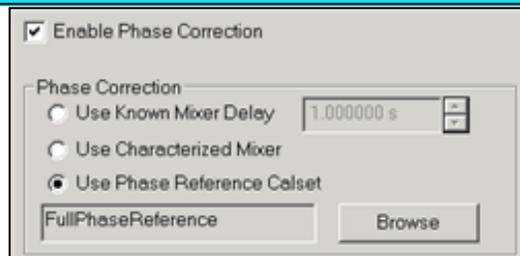
For this normalization, select the measurement point that has the best signal-to-noise ratio. The phase at the selected point will always be zero. This selection applies for both the measurement and the calibration sweeps.

The Reference Point is fixed at the middle point when [segment sweep](#) is selected.

Programming Commands

2. **Select a Phase [Format](#)** for the SMC measurement: click **Response**, then **Format**, then phase, unwrapped phase, or group delay.
3. During SMC Cal, at the [Cal Setup dialog](#), select the Phase Correction method as follows:

SMC Cal Setup - Phase Correction dialog help



Enable Phase Correction Check to enable Phase measurements. This can also be done during the mixer setup.

Phase Correction

Choose one of the following methods to specify the known delay through the characterized mixer. With the first two methods, the phase delay through a Calibration Mixer is measured and compared to the known delay, either entered, or stored in an *.S2PX file.

- **Use Known Mixer Delay** Enter the fixed, known, delay through the calibration mixer.
- **Use characterized mixer** Select, then browse to the *.S2P file that characterizes the calibration mixer. Use an *.S2PX file when making segmented SMC+Phase measurements. [Learn more](#). Use either of the following two methods to characterize the Cal Mixer over the SMC measurement frequency range:
 1. Use the [Mixer Characterization Wizard](#). The Cal Mixer has the same requirements as the **VMC Cal Mixer**. [Learn more](#).
 2. In a calibrated VMC channel, measure the group delay of the calibration mixer, then save to an *.S2P or *.S2PX file. However, a characterized mixer is required to calibrate the VMC channel.

- **Use Receiver Characterization Calset** Select, then browse to the Phase Reference Cal Set that covers the frequency range of the current measurement. [Learn more about SMC with a Phase Reference.](#)

Measuring phase on Mixers below 55 MHz

Measuring phase on mixers below 55 MHz is challenging for the following two reasons:

1. The PNA-X couplers have significant losses at these low frequencies. The PNA-X automatically reduces the IFBW at low frequencies in order to compensate for the coupler roll-off. And with SMC+Phase, we recommend setting 100 averages in order to reduce the noise of the SC21 delay measurement. These two settings combine to make the measurement speed VERY slow.
 - **Recommendation:** When measuring a downconverter with an output frequency below 55 MHz, check the "Reversed Port 2 Coupler" checkbox on the [SMC Mixer Sweep](#) tab and reverse the Port 2 coupler for the DUT measurement. With this setup, the IFBW is no longer reduced when measuring the output power. This significantly speeds up the measurement without reducing accuracy.
2. The PNA-X has a significant number of [frequency bands](#) below 55 MHz. SMC+Phase must have multiple measurement points within every frequency band.
 - **Recommendation:** Select the number of data points to yield a point spacing of NO MORE than 200 kHz per point. [Segment Sweep](#) can be used so that the point spacing above 55 MHz is not as dense.

Comparing SMC+Phase with VMC Phase Measurements

SMC phase measurements do NOT require a reference mixer, and are therefore easier to make than VMC phase measurements. Also the SMC calibration mixer is only required to have a known delay value, although an S2P characterization file provides more accurate results. The [Phase Reference](#) method provides the highest accuracy phase measurements for mixer/converters.

- When measuring converters with an embedded LO, SMC with phase can provide results that are as stable and free from sweep-to-sweep jitter as VMC.
- When measuring converters with an external LO that is shared with the reference mixer (as shown in the [VMC Setup diagram](#)), VMC provides results that are more stable than SMC+Phase.

How to improve the stability of SMC+Phase measurements

Stable phase measurements are attained by increasing [Sweep Averaging](#), and sometimes lowering the IFBW, until you attain the desired compromise between sweep time and trace jitter (the amount of random phase change at a single data point). For SMC+Phase, the default IFBW is 10 kHz, and 1 average. During calibration, the Averaging factor is temporarily multiplied by 4 to ensure an accurate phase calibration.

The following procedure shows **how to view and improve phase jitter**:

1. Create an SMC+Phase channel (Click Response, Measurement Class).
2. Enable Phase with SMC (See above) On the phase trace (to follow) notice that the only point that has NO jitter is the data point that you selected as the Phase Reference point.
3. Change the measurement to **IPWR**: (Click Response, Measure, IPWR)
4. Change Format to Phase. (Click Response, Format, Phase)
5. Normalize the trace. (Click Marker/Analysis, Memory, Normalize) [Learn more about Normalization.](#)
6. Autoscale the trace. (Click Response, Scale, Autoscale).
7. Optionally monitor the jitter with [Trace Statistics](#) (Std Dev)
8. Increase Averaging and possibly lower IFBW to improve jitter. (Click Response, Avg, then Average and IFBW). **Use a MINIMUM of 10 sweep averages (never use Point Averaging).**
9. After the adjustments are made, change the measurement back to your measurement of interest.
10. When measuring a new DUT, restart Averaging.

How to Calibrate a 2-stage (LO) SMC+Phase Measurement

Note: The following discussion does NOT pertain when a [Phase Reference cal](#) is used to correct the SMC+Phase Measurement.

When calibrating a dual-stage SMC+Phase measurement for group delay using a characterized mixer, the channel setup requires frequency values for two LOs, but the characterized-thru mixer uses only one LO. The frequencies of LO1 and LO2 are different. There are two ways to overcome this challenge:

1. Before the calibration, set the LO that is provided by an external source to uncontrolled. Then manually set the frequency of this external source to the LO frequency that gives the same input and output frequencies, and the same sweep direction, as the dual-stage setup. Perform the calibration under this condition. After the calibration, return the LO to controlled so that its frequency will be properly set during the measurement of the DUT.
2. Configure a 1-stage mixer setup, with the LO set to the frequency that gives the same input and output frequencies and the same sweep direction as the dual-stage setup. Perform the calibration under this condition. Save the calibration data as a user calset. Configure the dual-stage case, and apply the 1-stage calibration.

Last modified:

13-Feb-2013 Added Unknown mixer (9.90)
Created separate Phase Ref Cal topic
14-Jan-2013 Added '+ Phase' to how to..
11-May-2012 New topic

SMC Phase Reference Calibration

A Phase Reference calibration is performed to simplify the SMC+Phase calibration process.

In this topic:

- [Features, Requirements, and Limitations](#)
- [How it Works](#)
- [Hardware Setup](#)
- [Phase Reference Calibration Summary](#)
- [How to Perform the Phase Reference Calibration](#)

Other FCA Topics

Features

- Greatly simplified phase measurements on mixers.
- A Reference Mixer or Calibration Mixer is **NOT** required.
- Works well with segmented sweeps, and mixers with 2-stage LOs.

Requirements

- One of the following Agilent Comb Generators (Phase Reference):
 - U9391C (26.5 GHz)
 - U9391F (50 GHz)
 - U9391G (67 GHz) - Receiver attenuation must be used.
- Power meter or USB power sensor.
- S-parameter Cal Kit (mechanical or ECal module).
- For measurements below 55 MHz, an additional 'Unknown mixer' is required. [Learn more.](#)
- For measurements between 50 GHz and 67 GHz, an additional high-pass filter is required. Two back-to-back Agilent V281A waveguide-to-coax adapters is recommended.

Limitations

- Available with opt. 083 (NOT with opt. 082).
- Swept LO measurements are NOT allowed.
- Lowest frequency for Phase Reference Cal is 10 MHz.

How it Works

A Phase Reference Cal is performed, saved, and later recalled during a SMC Calibration. This is sometimes referred to as a 'tier 1' calibration. Due to stability of the PNA, the Phase Reference Cal can be performed infrequently. It is typically performed over the full frequency range of the PNA or Phase Reference so that it can be applied to all SMC calibrations that will be needed in the future.

The Phase Reference is a comb generator which produces signals at the multiple of its input frequency. By driving it with the PNA 10 MHz reference output, the Phase Reference produces signals at multiples of 10 MHz with a flat phase response.

During the tier 1 calibration, the PNA port 2 (B) receiver measures the phase at each 10 MHz comb frequency in order to generate correction values. These correction values are stored to a Phase Reference Cal Set which can be recalled and applied to correct subsequent SMC+Phase measurements.

Although only the B receiver is phase calibrated, an S-parameter calibration is performed at specified test ports and used to transfer the characterization to other PNA receivers.

A power calibration is also performed on port 1 as part of the Phase Reference cal. It does NOT have to be performed again during the SMC Cal.

Unknown Mixer Calibration (extends the phase reference calibration below 55 MHz)

By itself, the phase reference can only be used to calibrate down to 55 MHz. With one additional unknown mixer connection, the phase reference calibration can be extended to the full frequency range of the PNA-X. The unknown mixer is also used to improve the quality of the phase reference calibration below 300 MHz.

With the unknown mixer process, the phase of the unknown mixer is measured as a down converter with the output frequencies set between 10 MHz and 600 MHz. By using the data acquired using the phase reference, the system is able to make produce calibrated measurements of the mixer above 300 MHz. The phase response of the unknown mixer is assumed to be linear, so the expected phase response of the mixer is extrapolated down to 10 MHz. The variation in the measurements versus the expected phase response is attributed to, and used to correct, the phase response of the PNA receivers from 10 MHz to 300 MHz.

Unknown Mixer Properties

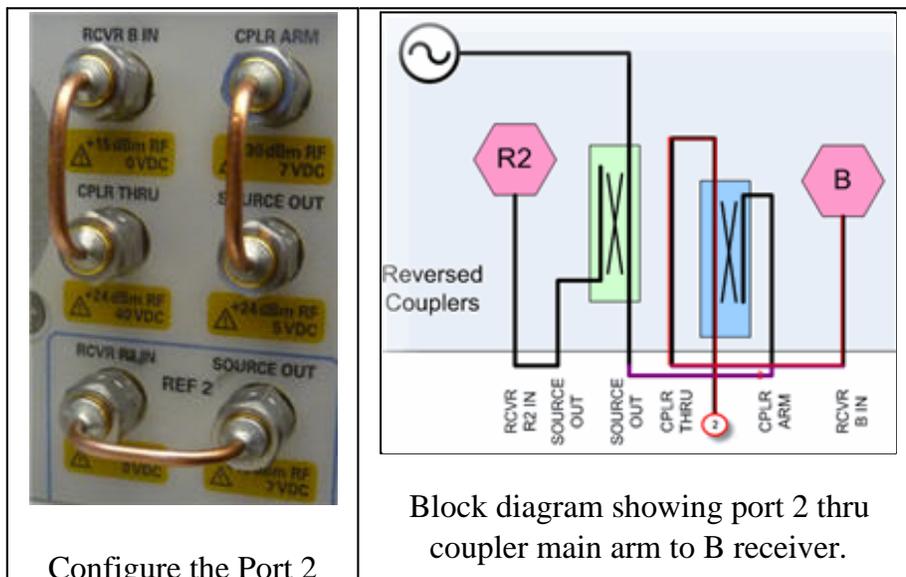
- The mixer must be able to output a signal from DC to 1GHz.
- Over these frequencies, the delay response of the mixer should be constant. Therefore, the unknown mixer must be passive, with no filtering or amplification.
- Add a 10dB attenuator on the mixer input and output to minimize mismatch effects.

Hardware Setup

1. Connect the U9391 Phase Reference to a DC power supply. See the [U9391 Technical Overview](#) for more information.
 - a. The green wire is "ground".
 - b. The red wire is + 15Vdc. The U9391C and U9391F only require +12 Vdc.
 - c. The black wire is not used.
2. Connect a cable from the PNA rear-panel 10 MHz Reference signal output to the U9391 input. The cable should be as short as possible.

Note: The U9391-60009 sine to square wave converter is **REQUIRED** for U9391 output frequencies above 26.5 GHz. See the [U9391 Technical Overview](#) for more information.

3. Connect the U9391 output directly to the PNA test port 2.
4. Insert the U9391 USB connector into any of the PNA USB ports.
5. Because of its low-frequency roll off, the coupler on Port 2 must be reversed which bypasses the coupled arm. Flip the port 2 front loops as shown in the following image. This increases power to the receiver by approximately 16 dB at low frequencies and removes noise from the measurement.



'B' receiver port front-panel loops to a vertical orientation as shown here.	
--	--

Phase Reference Calibration Summary

The Phase Reference Cal Wizard steps will show the appropriate prompts.

When Unknown Mixer is enabled (Start Frequency is 10 MHz)

- Reverse the Port 2 loops.
- The input and output match of the unknown mixer is measured. Then SC21 using 100 averages.

Phase Reference measurement and S-Parameter calibration below 50 GHz.

- S-parameter cal is NOT performed when Omit Coupler Measurements is checked.
- Reverse the Port 2 loops.
- When using the 67 GHz phase reference, use receiver attenuator or external attenuator.

When mixer output is above 50 GHz, Phase Reference measurement and S-Parameter calibration.

- Connect 50 GHz high pass filter to phase reference output.
- Reverse the Port 2 loops.
- Receiver attenuator or external attenuator is NOT used.

Full band power sensor measurement.

- Connect Power Sensor to Port 1.
- Port 2 coupler in normal position.

Full band S-parameter calibration.

- Connect standards to Port 1.
- Port 2 coupler in normal position.
- High-pass filter is NOT used.

- Receiver attenuator or external attenuator is NOT used.

How to start the Phase Reference Calibration

With an **Scalar Mixer / Converter** + **Phase** measurement active...

Using front-panel HARDKEY [softkey] buttons

1. Press **CAL**
2. then **[Start Cal]**
3. then **[Phase Reference Wizard]**

Using a mouse with PNA Menus

1. Click **Response**
2. then **Cal**
3. then **Start Cal**
4. then **Phase Reference Wizard**

Programming Commands

Phase Reference Cal Settings dialog box help

Use Unknown Mixer

Check to use an Unknown Mixer to calibrate the PNA receivers below 55 MHz. The Start Frequency becomes 10 MHz and can not be changed. [Learn how it works.](#)

Ports

- Ports 1 and 2 are always selected.
- Select port 3 and 4 to perform an S-parameter (SOLT) cal at those ports. The phase correction is transferred to all checked ports.

Omit Coupler Measurements

This choice can be selected when the stop frequency is less than or equal to 50 GHz. Enabling the checkbox simplifies the calibration procedure by reducing the number of calibration steps required, with some tradeoffs. Clearing the checkbox will always result in the most accurate phase-reference

calibration.

With the checkbox selected:

- Port 2 can NOT be used as the input to the DUT when setting up subsequent SMC+Phase measurements
- The power-sensor calibration is performed at the end of a test cable connected to Port 1, which may degrade the accuracy of conversion loss/gain measurements due to cable movement during the calibration. This problem can be minimized by using a high-quality cable and avoiding cable movement between calibration steps.

Start / Stop Frequency

Select the start and stop frequency of the Phase Reference Cal. Subsequent SMC+Phase calibrations MUST be the same or a subset of these frequencies. Interpolation is performed when the 10 MHz 'grid' of characterized frequencies is off from measurement frequencies.

- When Unknown Mixer is NOT checked, the lowest start frequency is **55 MHz**. The lowest stop frequency is 1 GHz.
- When Unknown Mixer IS checked, the start frequency is **10 MHz** and can NOT be changed. The lowest stop frequency is 5 GHz.

Source Attenuation

Select the attenuation value for the PNA source port.

Important Source Attenuation Notes

- This setting should match the port 1 source attenuator setting used for subsequent SMC+Phase measurements. If the settings are the same, only an S-parameter cal need be performed with the SMC+Phase calibration wizard to complete the calibration process. If the settings are different, a [Cal All calibration](#) must be performed, which requires both an S-parameter cal and another power-sensor cal. As an alternative, if you plan to make SMC+Phase measurements with different input attenuator settings, you may want to perform a couple of Phase Reference cals using these different attenuator settings.
- The phase reference cal is not as good when using input attenuation. You will get a better calibration if you use 0 dB of source attenuation. Then use the [Cal All](#) feature to perform the 2nd tier calibration with the required input attenuation. This is because the Cal All feature includes an attenuator calibration that will properly move the phase reference cal from the 0 dB attenuator plane to the new (required) attenuator plane.

Phase Reference

When the Phase Reference is detected by the PNA, it should appear in this field. If it does NOT appear here, try a different USB port, then click **Refresh**.

Cal Kit Connector

Select the connector type and gender of your Cal Kit.

Only ONE connector type and gender is specified. This is because the entire Phase Reference cal is performed at the PNA test ports to reduce the effects of cable flex in the characterization.

There may be times when an adapter or connector-savers are used to connect the phase reference and cal standards to the test ports. In these cases, for highest accuracy use that adapter style and gender for ALL connections to ALL test ports. The effects of the adapter will be de-embedded automatically during subsequent SMC+Phase calibrations.

Note: For highest accuracy, perform the phase reference cal at the test ports.

Cal Kit Select the Cal Kit that will be used to perform the S-parameter Cal.

B Receiver Attenuation

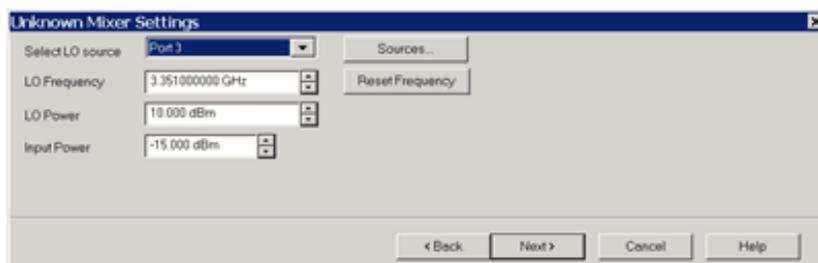
The 67 GHz (U9391G) phase reference outputs too much power and overloads the PNA Port 2 (B) receiver. Use at least 16 dB and not more than 20 dB of receiver attenuation to make accurate measurements using that phase reference.

This message is shown when a 67 GHz phase reference is connected to USB.

The B receiver attenuator control is shown if your PNA has receiver attenuators. Otherwise, connect external attenuation to the U9391G output when prompted.

Unknown Mixer Settings (Phase Reference Cal Wizard) dialog box help

The following wizard pages appear when Unknown Mixer is checked on the previous page:



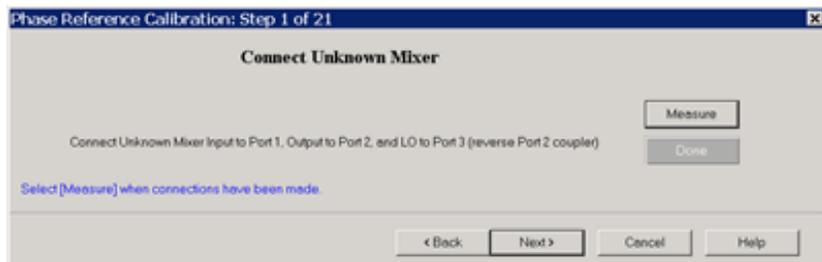
Select LO source The LO can be an internal 2nd source or an external source.

Sources Click to start the [External Devices dialog](#) where you can select or configure an external RF source.

LO Frequency Select a frequency that results in the unknown mixer output of 10 MHz to 600

MHz using this formula: $\text{Input} = \text{LO} + \text{Output}$.

Note: For best results, use the default LO frequency which avoids the PNA-X input (source) frequency band crossings.



The Port 2 coupler must be reversed for these measurements. [Learn how.](#)

- Connect the unknown mixer input to the between port 1 and port 2. For best results, connect the mixer output as close as possible to port 2.
- Connect the LO of the unknown mixer.
- Click **Measure**.
 - The input and output match of the unknown mixer is measured, then, SC21 using an averaging count of 100.
 - The data point spacing of the calibration is decreased to 156 kHz. This point spacing is necessary to accurately characterize all the low frequency band breaks of the PNA-X.

Power Cal Settings (Phase Reference Cal Wizard) dialog box help

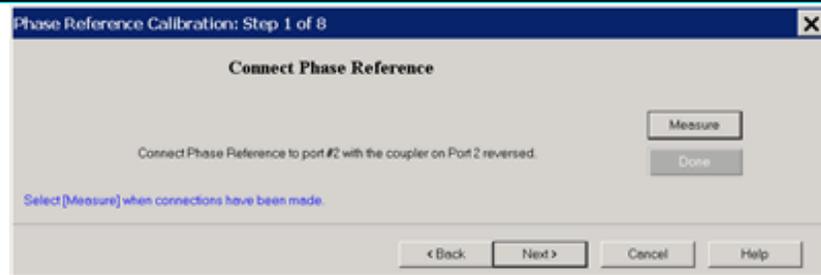


A power calibration is performed on port 1. This is done to simplify subsequent SMC calibrations since the power calibration will not need to be repeated.

For highest accuracy, connect the power sensor directly to the test port with no adapter.

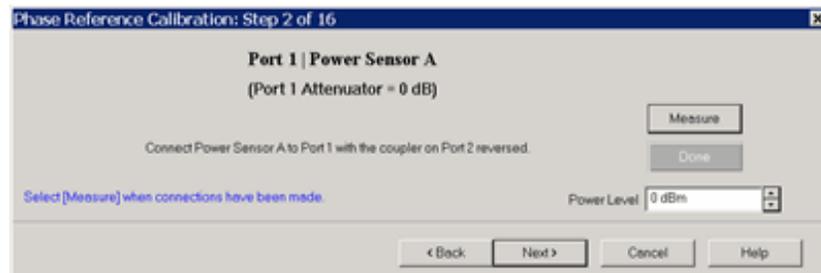
[Learn more about these settings.](#)

Phase Reference Calibration Steps dialog box help



1. Connect the Phase Reference directly to the port 2 test port.
 - a. If an adapter is necessary, then that adapter must be left in place for the SOLT cal at port 2.
 - b. When using a 67 GHz phase reference, attenuation or high-pass filter is required.
2. Ensure that the port 2 coupler is reversed. [Learn how.](#)
3. Press **Measure**.
4. The PNA will make a series of measurements which can take several minutes:
 - a. First, the match of the phase reference is measured.
 - b. Then, a sweep is made at multiples of 10 MHz across the entire frequency span. Each measurement is repeated 100 times and averaged to reduce the noise in the measurement.
 - c. An error is reported if the approximate power level is not detected.
5. When finished, press **Next>**

Power sensor A

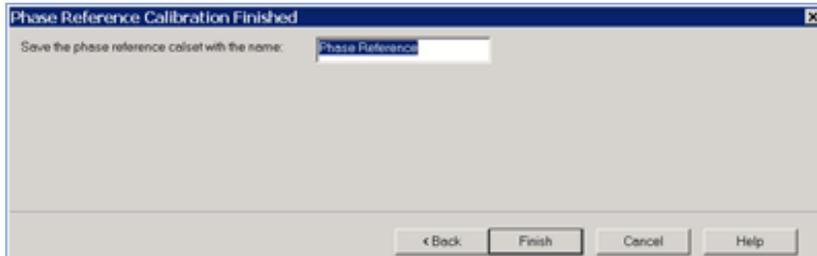


Connect the power sensor to port 1 test port.

SOLT cals are performed at the test ports.

- The THRU standard always performs an [Unknown Thru](#).
- Follow the prompts to complete the Phase Reference Cal.

Phase Ref Cal Finished



Enter a name for the Phase Ref Cal, then click **Finished**.

During subsequent SMC calibrations, select the Phase Reference Cal Set at the SMC Cal setup dialog. [Learn more](#).

Specify Delay



If you know the delay value of the Unknown Thru connection, enter it here.

Otherwise, click **OK** to accept the calculated value.

Last modified:

28-Oct-2013 Modified Omit Coupler DB

13-Feb-2013 Added Unknown mixer (9.90)

Created separate Phase Ref Cal topic

How to make an SMC Fixed Output Measurement

The following is a step-by-step example illustrating how to measure a 1-stage mixer in swept LO mode using FCA Scalar Mixer Calibration.

There are fewer components required for SMC as compared to [VMC](#), and fewer measurement steps. You can now make [relative phase measurements](#) with SMC. Also, ONLY SMC (not VMC) can measure the reverse conversion loss of the mixer.

This procedure can also be used for making **fixed** LO measurements, which is quite similar. Although a second source is still required, when using an external source, the physical triggering cables between the PNA and External Source are not required.

Required Equipment

- PNA-X or PNA 'C' models
 - with option 083 (FCA) or option 082 (SMC)
 - with PNA Rev. A.09.33 or above.

Note: This example has been updated to show keystrokes for the FCA User Interface available with PNA Rev. A.09.33 and above.

- GPIB External Source. Not necessary when using PNA-X with Internal Second source.
- ECal module with connectors that match the Input and Output connectors of the DUT. You can use adapters to make the ECal module match the DUT connectors, but first perform an [ECal user-characterization](#) with the adapters attached. ECal makes the FCA calibration much easier.
- GPIB or USB power meter / sensor
- Cables and adapters

Note: This procedure refers to an External Source to control the LO. If using a PNA-X with an [Internal second source](#), an external source is not necessary. Connect the LO directly to the second source output.

The example mixer

The example device is a down-converter mixer with the following characteristics:

- LO and Input Frequency Range: 2 GHz to 4.2 GHz

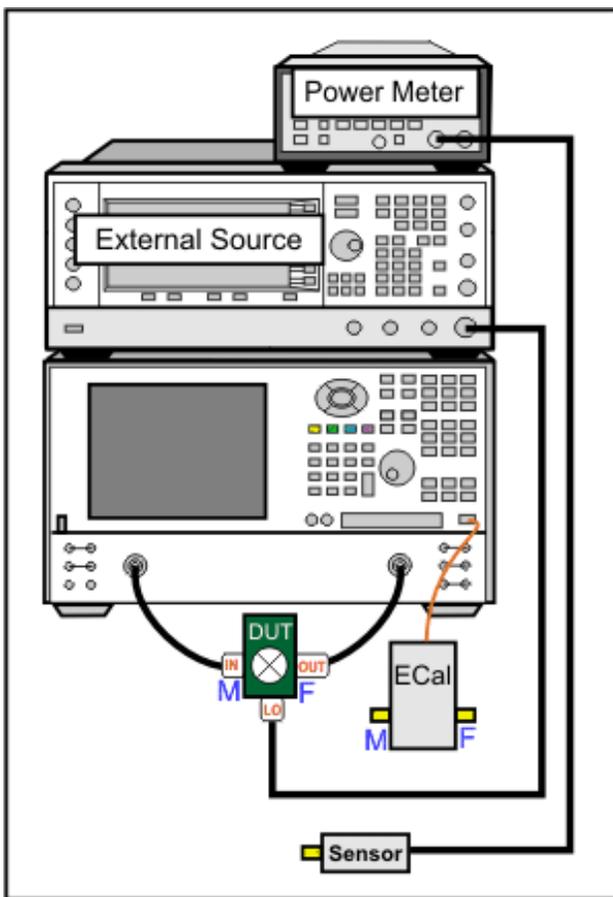
- Output Frequency Range: DC to 1.3 GHz

We will measure:

- Fwd Conversion Loss (SC21)
- Input Match (S11)
- Output Match (S22)
- Reverse Conversion Loss (SC12)

SMC Setup

Connect the devices as shown in the following diagram:



The DUT can be connected to any PNA ports. [Learn more.](#)

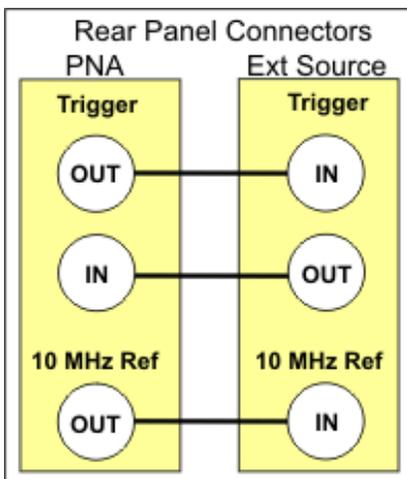
This procedure uses DUT input to PNA port 1 and output to PNA port 2.

This procedure refers to an External Source to control the LO. If using a PNA-X with an [Internal second source](#), an external source is not necessary. Connect the LO directly to the second source

output (port 3 or port 4 on a PNA-X).

Make Connections on the rear panels:

1. If using a GPIB power meter, connect the power meter to the external source GPIB connector. If using a USB power meter, connect it to any unused USB port.
2. If using a PNA-X with an [Internal second source](#), the following three steps are not necessary. Connect the LO directly to the second source output.
3. Using a GPIB cable, connect the PNA GPIB controller port to the external source GPIB connector.
4. Using two BNC cables, connect the Source and PNA Trigger connectors as shown in the following image. This is not necessary when making fixed LO measurements.
5. Using a BNC cable, connect the PNA **10 MHz Reference Output** to the Ext. Source **10 MHz Reference Input**.



Create the Measurement

For this document:

- Front-panel hardkeys are formatted as "Press **TRACE**"
 - Front-panel softkeys are formatted as "Press [**S11**]"
 - Menus are formatted as "Click **System**"
1. Connect the DUT.
 2. On the PNA, press System, then [**Configure**], then [**Power Meter Settings**].
 3. Under Interface, select **GPIB**, then enter the power meter address. Or select **USB**, then select the USB power meter that is connected to the PNA.

4. Press **Preset** to make sure you are starting with a known state.
5. Press **Meas** then **[Measurement Class]**, then **Scalar Mixer/Converter**, then **OK**. At the **Confirm...** dialog, click **OK**. An S11 trace is created.
6. Press **[SC21]** to replace the S11 trace.

Configure the Mixer settings

1. Press **Freq** then **Input** to start the Mixer Setup dialog.
2. On the Mixer Frequency tab, enter the Mixer setup values as shown in the image below.

Input	Start/Stop	2.579500000 GHz	3.580500000 GHz	Calc Input
LO1	Start/Stop	1.779500000 GHz	2.780500000 GHz	<input checked="" type="checkbox"/> Input > LO
Output	Fixed	<input type="radio"/> +	4.359000000 GHz	Calc LO
		<input checked="" type="radio"/> -	800.000000 MHz	Calc Output

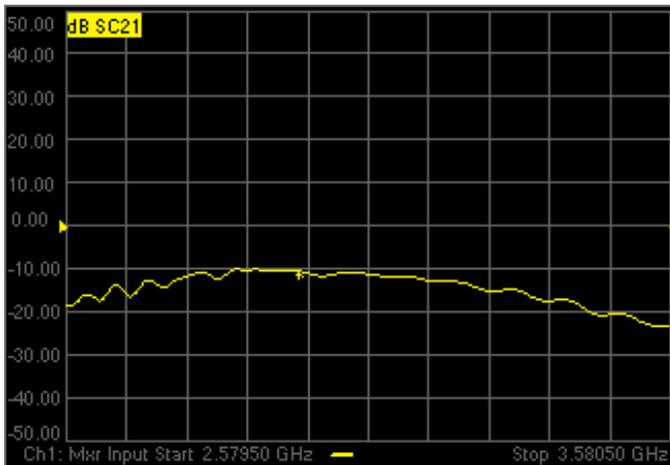
Notes:

- To provide quicker access to the Mixer Setup dialog, use the Setup softkey. [Learn how.](#)
- Rather than enter ALL of the frequency settings, you can enter the Input and the Output frequencies, then click **Calculate LO**.
- If **Input>LO** is NOT checked, the PNA assumes you want the Input < LO frequencies, and higher LO frequencies are calculated as a result.
- The LO power level setting specifies the power out of the external source; not at the DUT) unless an LO power cal is performed.
- When the settings are valid, the background color around the **Apply** button is available.

Configure the LO Source

1. On the Mixer Setup tab:
2. Change LO1 to either an internal PNA source or a pre-configured external source.
3. To configure an external source:
 - a. Click **Add Source** in the upper-right corner of the Mixer Setup dialog.
 - b. Complete the [External Source Configuration](#) dialog.
 - c. If there is a problem communicating with the source, the PNA will display an error. [See Problems?](#)

- d. Click **OK** to return to the Mixer Setup dialog.
 - e. The new external source is now available as an LO1 selection.
4. Save the mixer settings in a file so you can recall them easily. Click **Save...**, then type a descriptive filename, such as "FixedOutputMixer".
 5. The trace should begin to sweep as the external source steps in frequency. It should look something like this:



Problems?

Not sweeping:

- On the PNA, press Trigger, then **Continuous** to start the PNA sweeping. Watch for error messages on the PNA and source.

Problems communicating with the source:

- Press System then **Configure** then External Source. On the Select Sources dialog, click Configure. Select your LO source. Click **Software CW trigger**, then close the dialog. Again, press Trigger, then **Continuous** to start the PNA sweeping. If this works, then something is wrong with **Hardware (BNC)**. Check the trigger cables on the rear panel.
- Can the PNA communicate with the power meter? If not, there is something wrong with the GPIB or USB communication.
- As a last resort, try rebooting the PNA. First, [save the entire setup to a .csa file](#). When the PNA preset measurement appears, recall this .csa file and continue at this step.

If the source is sweeping, and the PNA Input is sweeping, but there is still no output.

- Check power levels at the LO and Input.
- Check the DUT by making a fixed LO measurement - much easier.

Tip: You can optionally calibrate the LO Power level at the DUT using a standard [Source Power Calibration](#). Select the source port in the Source Power Cal dialog.

Perform an SMC calibration

1. Disconnect the DUT.
2. Connect the ECal module to a PNA USB port.
3. Press Cal, then **[Start Cal]**, then **[Cal Wizard]**. Because the SC21 measurement is active, the Cal Wizard automatically begins an SMC calibration.
4. At the **Calibration Setup** dialog, click **Next**.
5. At the **Select DUT Connectors and Cal Kits** dialog, for **DUT Port 1** select the connector type and gender of your DUT INPUT. For **DUT Port 2** select the connector type and gender of your DUT OUTPUT. Then select ECal as the Cal Kit to use for each connector. Click **Next**.
6. At the **Scalar Mixer Calibration Step 1 of 2** dialog, connect the power sensor to the Port 1 test cable, then click **Measure**. The data will be used to correct for input mismatch errors.
7. At the **Scalar Mixer Calibration Step 2 of 2** dialog, connect the ECal module Port A to the Port 1 cable, and Port B to the Port 2 cable. Then click **Measure**. This portion of the calibration gathers the linear (non-frequency-translating) error terms of the test setup at the input and output frequencies.
8. At the **Calibration completed** dialog, you can choose to save the SMC calibration as a User Cal Set. Otherwise, click **Finish** to complete the SMC calibration. Correction is turned ON and applied to the SMC trace.

What is happening?

When an external source is sweeping, the measurements are much slower. When correction is ON, you will see that there are times when nothing is happening on the screen. This is because there are background measurements being made but not displayed.

This is exactly the same as when full 2-port correction is applied to an S-parameter. All four parameters are measured, then correction is applied, then all four measurements are updated. This occurs much faster when there is no external source. With correction OFF, the traces are updated as the data is measured. You can see this taking place by creating the following measurements.

Create S12 Upconverter, S11 Input and S22 Output Match

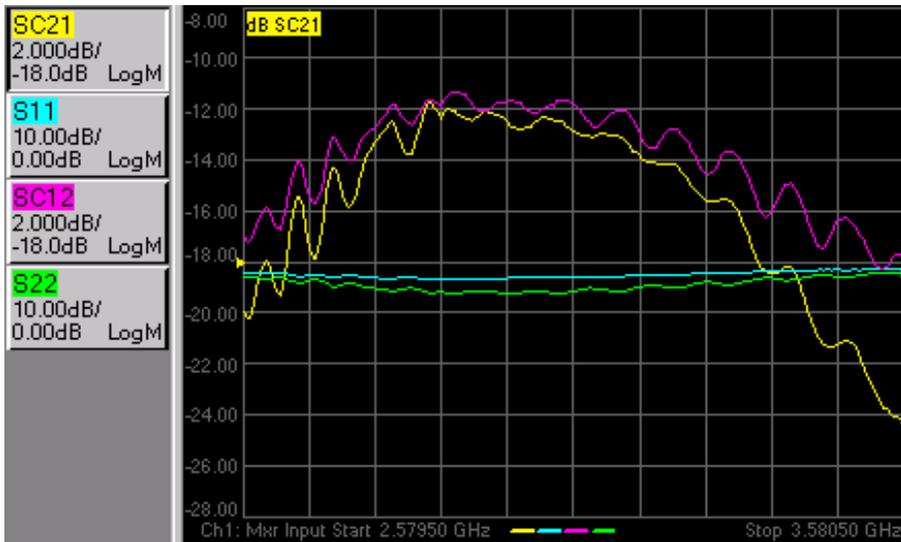
1. Press Meas then **[S11]**, **[SC12]**, and **[S22]** to add these measurements to the same channel.
2. While the source is sweeping, watch the source port indicator on the front of the PNA. First, the port 1

indicator will light for two sweeps, then the port 2 indicator will light for 2 sweeps. During the last sweep, all 4 traces update.

3. Press Cal, then **[Correction OFF]**. Notice that the relevant traces update as the sweep is occurring.

With the SC12 measurement you can see the reciprocity of the mixer.

Note: With the recent improvements to FCA, this step is MUCH easier than before. SMC forward and reverse measurements can now reside in the same channel and are calibrated automatically at the same time.



Last Modified:

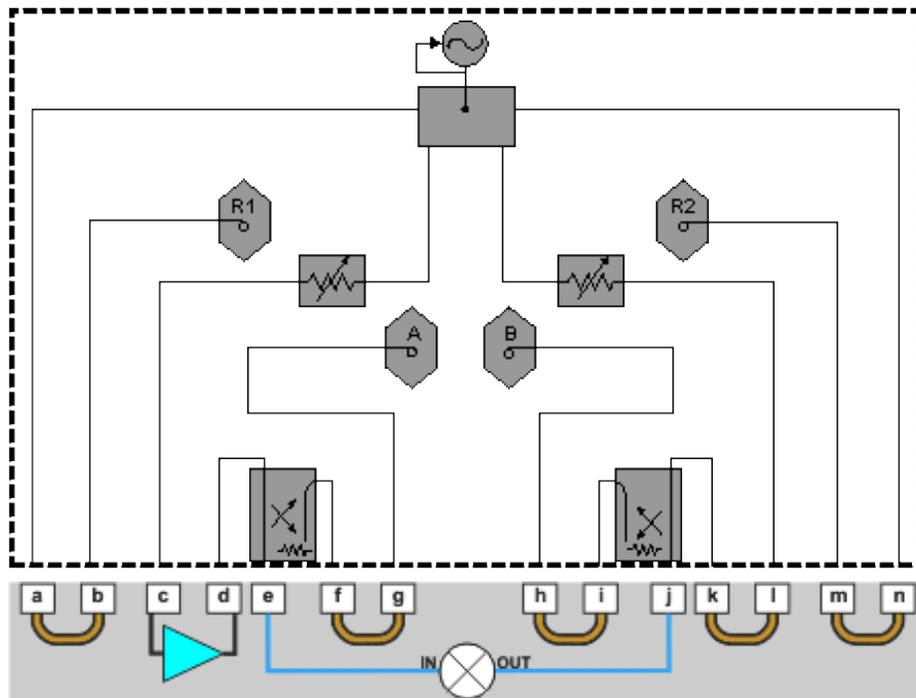
4-Apr-2011	Updated for A.09.33
14-Apr-2010	Added N5230A/C and Opt 082
6-Apr-2009	Updated for PNA-X
5-May-2008	Modified for selectable ports
17-Apr-2008	Added 10 MHz ref image

SMC with a Booster Amp

If your mixer measurement requires more source power on the input than the PNA can provide, a booster amplifier can be used to provide the additional power. This topic describes how to configure and make a calibrated SMC measurement using a booster amplifier.

Connect

Connect the booster amplifier between the Source-Out and Coupler-Thru connectors on the front-panel as shown in the following diagram. Your PNA block diagram may not look like this.



Item	Description	Item	Description
a	SOURCE OUT	h	RCVR B IN
b	RCVR R1 IN	i	CPLR ARM
c	SOURCE OUT	j	PORT 2
d	CPLR THRU	k	CPLR THRU
e	PORT 1	l	SOURCE OUT
f	CPLR ARM	m	RCVR R2 IN
g	RCVR A IN	n	SOURCE OUT

Measurement and Calibration Setup

In the following procedure:

- **Test Port power** is the power level out of the source.

- **Corrected power** is the power level you require at the mixer input and output.

This procedure assumes you will applying stimulus power to the mixer **input** to make SC11 and SC21 measurements, and to the **output** of the mixer to make SC22 and SC12 measurements.

1. Determine the gain of the booster amplifier. If the gain has significant slope across the **input and output range** of the mixer, see [Booster Amp with a Gain Slope](#).
2. Determine the corrected power for both the input (port 1) and output (port 2) of the mixer.
3. Calculate the Test Port power for both ports by subtracting the gain of the amplifier from both the input and output corrected power levels.

For example, the following values assume a 25 dB booster amp on port 1 as in the diagram above.

	Corrected Power	-	Amp Gain	=	Test Port Power
Port 1 (input)	0 dBm	-	25 dB	=	-25 dBm
Port 2 (output)	-20 dBm	-	25 dB	=	-45 dBm

4. On the PNA [Power dialog](#), clear the **Port Power Coupled** checkbox, which allows different power levels for each port.
5. Enter the calculated **Test Port Power** values for each port.
6. During the SMC Cal Wizard **Select DUT Connectors and Cal Kits** dialog, click **View/Modify Source Cal Settings** to invoke the [Source Calibration Settings dialog](#).
7. In **Power Offset**, enter the booster amplifier gain.

Booster Amp with a Gain Slope

SMC calibration takes place over the entire input and output range of the mixer. Therefore, the booster amplifier will also be subjected to the entire input and output frequency range of the mixer.

To compensate for a gain slope, you might have to experiment with the source attenuator setting, power-offset value, and initial power value to get a combination that will not cause the PNA source to go unlevelled during or after the cal.

For example, assume the booster amp gain is 30 dB at the low end, and 20 dB at the high end. If you enter 30 dB for the power offset value, the PNA might run out of ALC range when the actual gain drops to 20 dB. The PNA will try to increase its source power to account for the 10 dB gain drop. Therefore, pick a power offset value that is in the middle of the amplifier gain band (25dB).

If possible, select a PNA attenuator setting that puts the ALC approximately in the middle of its range at the desired corrected power with the mid-band gain. This condition means the ALC can set the power higher and lower to account for the gain slope, without unleveling.

If the gain slope is too large, then there may not be a setting that prevents a source unlevel. In this case, a flatter booster amp must be used.

Last Modified:

19-May-2008 Edited for generic image

VMC Measurements

VMC Setup and Calibration is very similar to SMC. See [FCA Overview](#) to learn about the features that are common to these two applications.

The following information is unique to VMC:

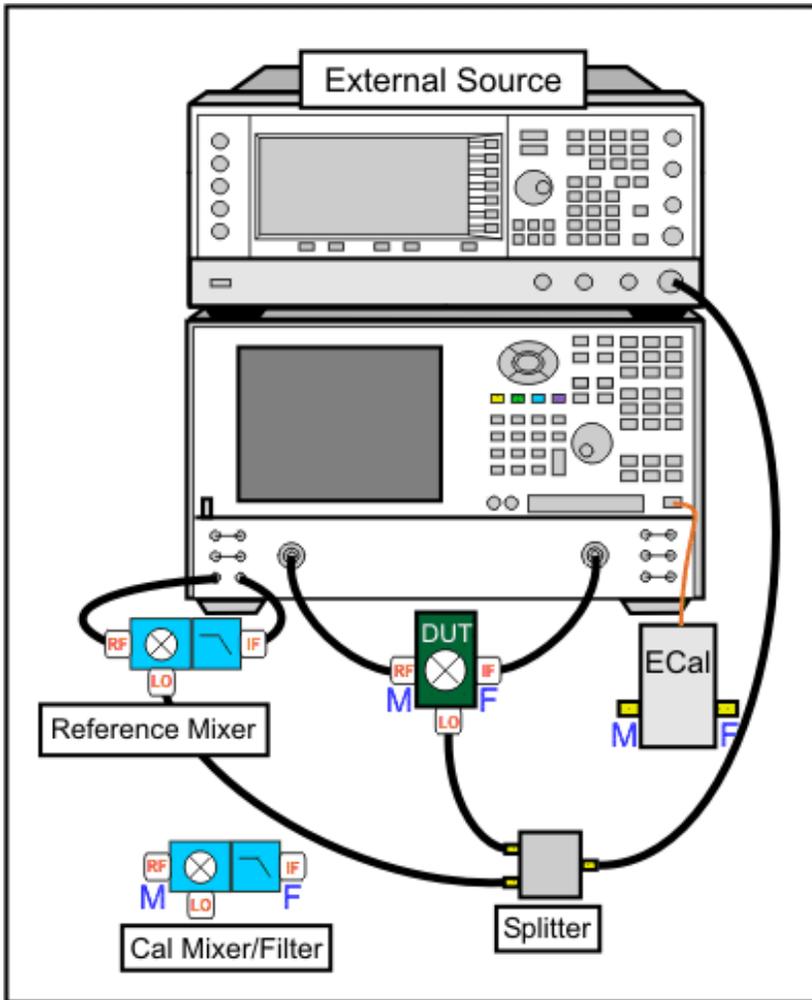
- [VMC Hardware Setup](#)
- [Create a VMC Measurement](#)
- [VMC Parameters Offered](#)
- [The VMC Mixer Setup dialog](#)
- [VMC Calibrations](#)

See Also

[Embedded LO](#)

[How to make a VMC Fixed Out measurement](#)

VMC Hardware Setup



DUT Input (RF) must be connected to PNA port 1.

DUT Output (IF) can be connected to any other PNA port.

Notes:

- When using a PNA-X with Internal Second Source, the external source is NOT necessary.
- [See note regarding LO power out both second source ports](#)
- [Learn which PNA ports can be used for the LO.](#)
- [Measure a DUT with an Embedded LO](#)

Reference Mixer

The Reference mixer provides a phase reference for the measurements. The reference mixer is connected in the reference receiver path of the network analyzer, between the source out and receiver R1 in ports, as shown below.

The reference mixer is considered part of the test system setup like the test cables. It remains in place during the entire calibration and measurement process. The reference mixer is switched in and out of the measurement path by the PNA as needed. [See how to manually switch the reference mixer.](#)

The reference mixer does not need to be reciprocal and does not have to match the calibration mixer or the mixer-under-test in performance. The only requirement of the reference mixer is that it cover the same frequency range as the mixer under-test. In general, it is valuable to select a reference mixer that can be used with a variety of different setups. For example, a broadband mixer can be used in place of several narrow-band alternatives.

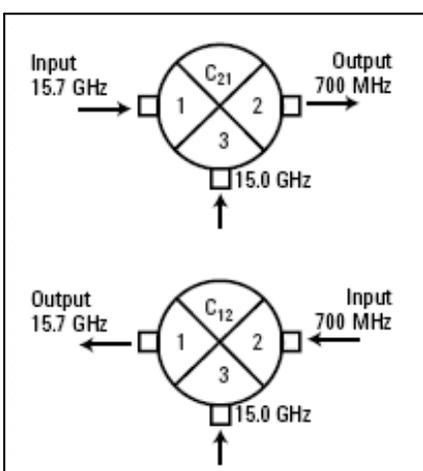
A low pass filter on the output of the reference mixer can be used to suppress the LO leakage signal that comes out of the reference mixer output. It is not strictly needed, but ensures that the PNA will not have any source unlock or unlevel errors due to the LO leakage.

- Connect the Reference Mixer INPUT to PNA **Ref 1 Source out**
- Connect the Reference Mixer OUTPUT to PNA **Rcvr R1 In**

Calibration Mixer/Filter

The Calibration mixer/filter is characterized either before or during a VMC calibration. It is used during the VMC calibration as the THRU standard. The calibration mixer/filter combination must meet the following requirements:

- The mixer must be reciprocal over the frequency range of the mixer under test. This means that it has the same magnitude and phase response in the up-converting and down-converting directions ($C_{21} = C_{12}$) as shown in the following diagram.



- If the Input and Output frequency ranges are overlapping, the mixer must have Input to Output Isolation greater than 10 dB more than the conversion loss in the overlapping range.

- The filter must reject the undesired mixing product, and pass the desired mixing product, at the output of the cal mixer. This requirement can be made easier by characterizing the mixer/filter as a downconverter. [Learn more.](#)

Note: With a corrected VC21Swept LO measurement, the phase data is displayed relative to the phase of the calibration mixer that was used during the VMC calibration. In addition, Group delay display format is NOT valid.

[See an example of a Fixed Output VMC Measurement](#)

Important note: Orientation of Reference mixer and Calibration mixer/filter

The reference mixer is ALWAYS connected in the same orientation as the DUT, since the output frequency of the reference mixer has to match that of the DUT. The same applies to the calibration mixer/filter if it is characterized as part of a full VMC cal.

If you characterize the calibration mixer/filter separately, you can characterize it as either an upconverter or downconverter. [Learn more.](#)

LO Source

Note: When using a PNA-X with Internal Second Source, the external source is NOT necessary.

- [See note regarding LO power out both second source ports](#)
- [Learn which PNA ports can be used for the LO.](#)
- Connect **External Sources** to the PNA GPIB Controller port.
- Learn how to [Configure an External LO Source](#)

Create a VMC Measurement

1. On the PNA-X front panel, press **Meas** then **[Measurement Class]**
2. Select **VMC**, then either:
 - **OK** delete the existing measurement, or
 - **New Channel** to create the measurement in a new channel.
3. A VC21 measurement is displayed.
4. To select additional parameters to display, click **Trace**, then **New Trace**, then select a parameter from the list.

VMC Parameters Offered

Important Note: Connecting your DUT to the PNA:

RF and **IF** terminology is NOT used in FCA because the PNA does not know how the DUT is labeled or how it will be used. Instead, the general terms INPUT and OUTPUT are used.

- **INPUT** - The DUT port being stimulated with frequencies before conversion.
- **OUTPUT** - The DUT port outputting converted frequencies.

INPUT and OUTPUT Frequencies are specified using the [Mixer Setup dialog box](#).

The DUT input is always connected to PNA port 1. However, the DUT output can be connected to any other PNA port.

- **VC21, VC31, or VC41 Conversion Loss/Gain (default)** - stimulus at Input, response at Output
- **S11** - stimulus and response at Input
- **S22, S33, or S44** - stimulus and response at Output
- **R1 (or R)** - stimulus at Input, measures absolute power at the R1 receiver (uncorrected)
- **B, C, or D** - stimulus at Input, measures absolute power at the output receiver (uncorrected)
- Reverse conversion loss is NOT offered because of the reference mixer.

See Also

[Measure a DUT with an Embedded LO](#)

VMC Mixer Setup

How to start the VMC Mixer Setup dialog

To provide quicker access, use the Setup softkey. [Learn how](#).

Using front-panel HARDKEY [softkey] buttons

1. Press **FREQ**
2. then [**Mixer Setup**]

Using a mouse with PNA Menus

1. Click **Stimulus**
2. then **Freq**
3. then **Mixer Setup**

[Programming Commands](#)

The following VMC Mixer Setup dialog tabs are presented:

- [Sweep Tab](#) (shared with SMC)
- [Power Tab](#) (shared with SMC)
- [Mixer Freq Tab](#) (shared with all converter apps)
- [Mixer Power Tab](#) (shared with all converter apps)
- [Mixer Setup Tab](#) (shared with all converter apps)

VMC Calibration Overview

The Calibration Wizard guides you through this process. The first three steps characterize the calibration mixer that is used as the THRU standard during the calibration process.

1. Perform a [2-port SOLT calibration](#) over the INPUT frequency range of the DUT, and another [2-port SOLT calibration](#) over the OUTPUT frequency range. Use either a mechanical calibration kit or an ECal module.
2. Characterize the input and output match of the [calibration mixer/filter combination](#) with the external LO connected and the output terminated with an open, short, and load. [Learn how to connect the calibration mixer/filter](#). Once characterized, an S2P file is saved and can be recalled for use in subsequent VMC calibrations using the same stimulus settings.
Note: Use an *.S2PX file for SEGMENTED VMC measurements. [Learn more](#).
3. Connect the [reference mixer](#) between the Source Out and Rcvr R1 front-panel connectors. Connect the output port of the calibration mixer/filter combination to PNA Port 2 (or at the end of the cable attached to the port).
4. Measure the calibration mixer/filter combination as the THRU calibration standard.
5. The PNA calculates the error terms necessary to make corrected phase measurements of your mixer/converter under test.

VMC Cal Wizard

The following dialog boxes are presented during VMC Calibration and VMC [Mixer Characterization](#).

- [Calibration Setup](#)
 - [Waveguide/In-fixture/On-Wafer Setup](#)
- [Calibration Mixer Characterization](#)
 - [Measurement Direction](#)
- [Select DUT Connectors and Cal Kits](#)
 - [Modify Frequency Cal](#)
 - [Specify how the ECal module is connected](#)
 - [Modify Mixer Cal](#)
 - [Select the ECal Port to be connected to the Output of the Calibration Mixer](#)
- [Vector Mixer Cal Steps](#)
- [Measure Calibration Standards](#)
 - [Save Mixer Characterization](#)
- [Calibration Completed](#)
- [Specify Adapter Delay](#)

How to Perform a VMC Calibration

1. [Create an FCA measurement](#), then...

Using front-panel HARDKEY [softkey] buttons

1. Press CAL
2. then [**Start Cal**]
3. then [**Cal Wizard**]

Using a mouse with PNA Menus

1. Click **Response**
2. then **Cal Wizard**

To perform Mixer Characterization ONLY

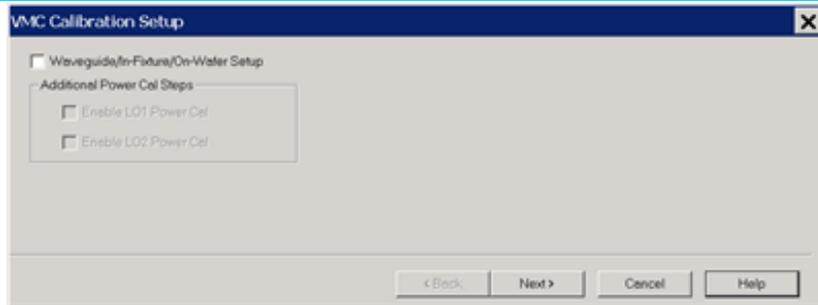
1. Not Available

1. Click **response**

2. then **Cal**
3. then **Start Cal**
4. then **Mixer Characterization Wizard**

Programming Commands

Calibration Setup dialog box help

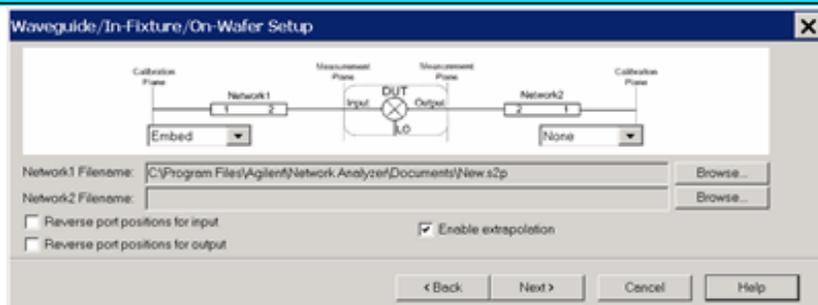


Waveguide/In-fixture/On-Wafer Setup Check to embed or de-embed circuit networks on the input and output of your mixer under test. Starts the following dialog box.

Additional Power Cal Steps

Enable LO1 / LO2 Power Cal Check when LO1 / LO2 is controlled (on the [Mixer Setup](#) tab) to perform a Power Cal on the LO source(s).

Waveguide/In-fixture/On-Wafer Setup dialog box help



This dialog box appears ONLY if you checked the **Waveguide/In-fixture/On-Wafer Setup** box in the previous [Cal Setup](#) dialog.

Allows you to embed or de-embed circuit networks on the input and output of your mixer under test.

For Network1 (Input) and Network2 (Output) select **Embed**, **De-embed**, or **None**.

Browse Click to navigate to the .S2P file that models the network to embed or de-embed.

Reverse port positions for input/output Check to cause the Fixture/Adapter to be configured with Port 2 connected to the PNA and Port 1 to be connected to the DUT. The image in the dialog is updated to reflect that change.

Enable Extrapolation Check (default setting) to apply a simple extrapolation when the S2P file has a narrower frequency range than the channel. The values for the first and last data points are extended in either direction to cover the frequency range of the measurement. A warning message is also displayed when extrapolation is necessary.

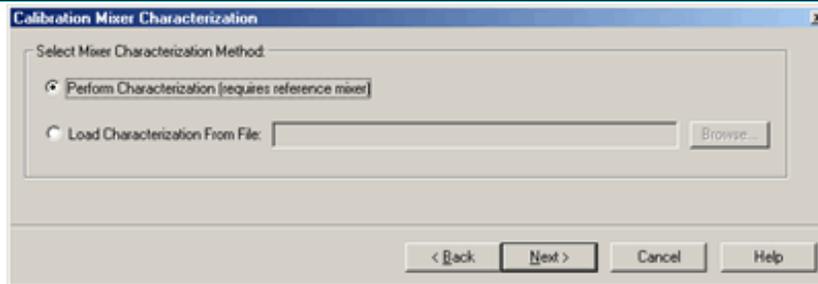
To Embed or De-embed

- When you have a 2 port network that needs to be connected between the Cal reference plane and the DUT during the measurement, but it is NOT present during the calibration, then that network has to be **De-Embedded** from the port in question during the calibration. In other words, De-Embedding in FCA calibration extends the calibration reference plane to include the two port network.
- When you have a 2 port network that is included as part of the calibration reference plane but has to be disconnected during the measurement, then that 2-port network has to be **Embedded** for the port in question during the calibration. In other words, Embedding in FCA calibration retracts the calibration reference plane to exclude the two port network during the measurement.

Notes

- [Characterize Adaptor Macro](#) can be used to create the S2P file.
- Interpolation is performed when more frequencies are included in the file than in the channel, and the data points do not exactly match those of the measurement.

Calibration Mixer Characterization dialog box help



What is Calibration Mixer Characterization? For a brief explanation, see [Calibration Mixer](#).

Select Mixer Characterization Method

Perform Characterization (requires a reference mixer) Performs a Mixer characterization in addition to the VMC calibration. The mixer characterization file will be saved at the end for use in subsequent VMC calibrations. Choose this selection if you do NOT already have a mixer characterization file to load.

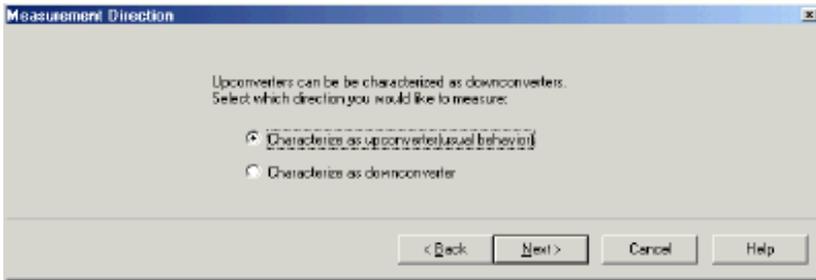
Load characterization from file Loads an S2P calibration mixer characterization file. Click **Browse** to locate the file.

Note: Load an ***.S2PX** file for SEGMENTED VMC measurements. [Learn more](#).

- The frequency range of the S2P file **MUST** be the same, or larger than, the frequency range of the FCA measurement. If the S2P file frequency range is larger, or the data points do not exactly match those of the measurement, interpolation will be performed.
- The VMC calibration requires that the calibration mixer be connected in the same orientation as that in which it was characterized. The direction in which it was characterized is not part of the file that is recalled. You have to remember and connect it appropriately.

"Invalid Mixer Characterization File" is displayed if the frequency range of the S2P file is smaller than those of the measurement.

Note: A Mixer Characterization Cal can be performed separately. [Learn how](#).



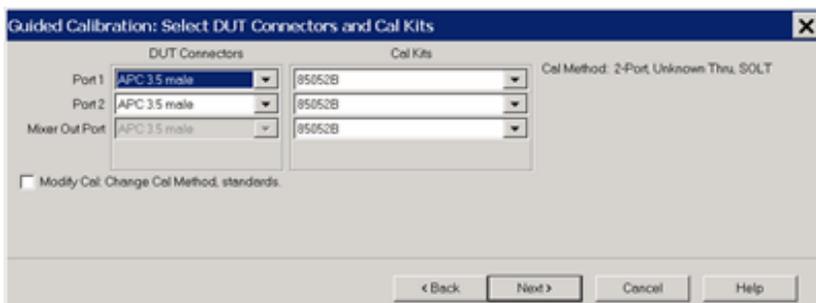
Measurement Direction dialog box help

This dialog box appears ONLY if your settings in the [Mixer Setup](#) dialog box indicate that your DUT is being tested as an upconverter (input < output). It allows you to characterize the [Calibration Mixer / Filter](#) as a downconverter (input > output) or an upconverter.

The following example shows why you would choose to characterize the calibration mixer as a downconverter. Consider a DUT being used as an upconverter. The input frequency is 70 MHz, the LO is 20 GHz, and the selected (+) output frequency is 20.07 GHz. If we chose (-) in the [mixer setup dialog](#), the output frequency would be 19.93 GHz.

- **Characterize as upconverter** A very sharp cutoff filter is required to reject the undesired output of 19.93 GHz and pass the desired 20.07 GHz.
- **Characterize as downconverter** The input frequency is 20.07 GHz; the LO is 20 GHz. The sum (+) output is 40.07 GHz and the diff (-) output is 70 MHz. These are very easy to separate with a low-pass filter. The original frequencies are always used in the downconversion process, so be sure to choose a filter that will pass 70 MHz and reject 40.07 GHz.

[See connection diagrams.](#)



Select DUT Connectors and Cal Kits dialog box help

Allows you to specify the connector type and Cal Kit for each DUT port.

Port n For each listed PNA port, specify the DUT connector type and gender, and the Cal Kit to use.

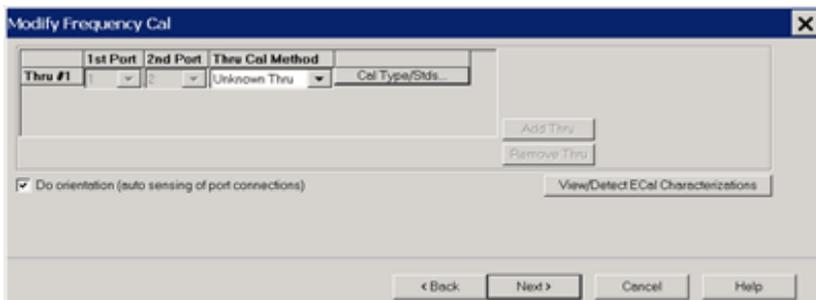
Mixer Out Port Output port of the image filter that is connected to the [calibration mixer](#). Specify the Cal Kit / standards to use for the measurement of the calibration mixer / filter combination.

Note: When selecting a cal kit for the Mixer Out Port, be sure that the kit has standards with connectors that can mate to the mixer output port. If you choose an ECal, the ECal must have at least one port that can mate to the mixer output port.

Note: If your DUT connectors are:

- **Waveguide** Change the system impedance to 1 ohm before performing a calibration. See [Setting System Impedance](#).
- **Not listed** (male and female) Select **Type A** as the connector type. Type A requires a calibration kit file containing the electrical properties of the standards used for calibration (see [Calibration kits](#)).
- **Unspecified** (like a packaged device) Select **Type B** as the connector type. Type B requires a calibration kit file containing the electrical properties of the standards used for calibration (see [Calibration kits](#)).

Modify Cal Check to start the [Modify Cal](#) dialog. If performing a Mixer Characterization Cal at the same time as VMC Cal, two Modify Cal dialogs will be presented, one after the other.



Modify Frequency Cal dialog box help

For VMC calibrations - NOT for Mixer Characterization.

Thru Cal Method For each Thru connection, choose the Thru method. [Learn more about these choices.](#)

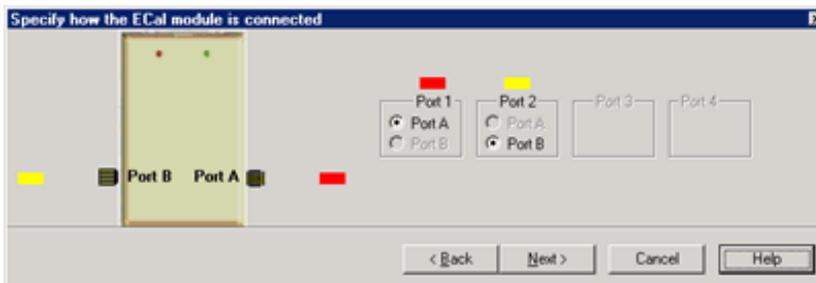
Cal Type/Std Click to start the [Modify Calibration Selections](#) dialog box.

The following selections are available ONLY if using an ECal module.

Do orientation When this box is checked (default) the PNA senses the ECal model and direction in which the ECal module port is connected to the PNA ports. If power to the ECal module is too low, it will appear as if there is no ECal module connected. If you use low power and are having this problem, clear this check box to provide the orientation manually.

Orientation occurs first at the middle of the frequency range that you are calibrating. If a signal is not detected, it tries again at the lowest frequency in the range.

View/Detect ECal Characterizations Appears only if an ECal module is selected for use. Click to invoke the [View ECal Modules and Characterizations](#) dialog box. Displays a list of ECal modules that are connected to the PNA.



Specify how the ECal module is connected dialog box help

This dialog box appears when the **Do orientation** checkbox in the previous **Modify Frequency** dialog box is cleared.

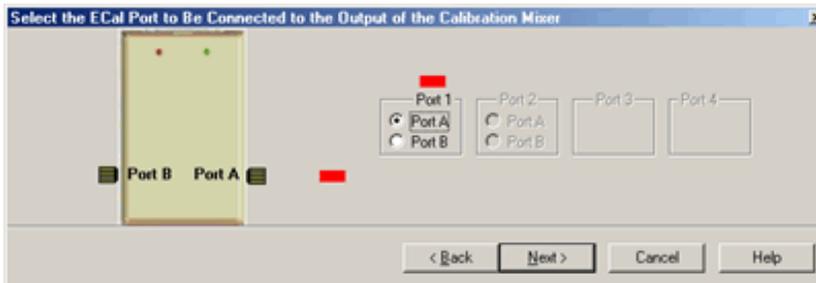
Click the ECal Port that is connected to each PNA port.



Modify Mixer Cal dialog box help

Mixer Characterization ONLY. The Thru standard is not measured. Therefore, the Thru Cal Method choices are not available.

View / Detect ECal Characterizations Available ONLY if using an ECal module. Invokes the [Select ECal Module and Characterization](#) dialog box.



Select the ECal Port to be connected to the Output of the Calibration Mixer dialog box help

Select the ECal Port to be connected to the output of the image filter of the Calibration Mixer / Filter combination. [See connection diagram](#) of Calibration Mixer / Filter combination.



Measure Calibration Standards dialog box help

Prompts for standards to be measured. Connect the standard, then click **Measure**.

Measure Measures the mechanical standard and continue to the next calibration step.

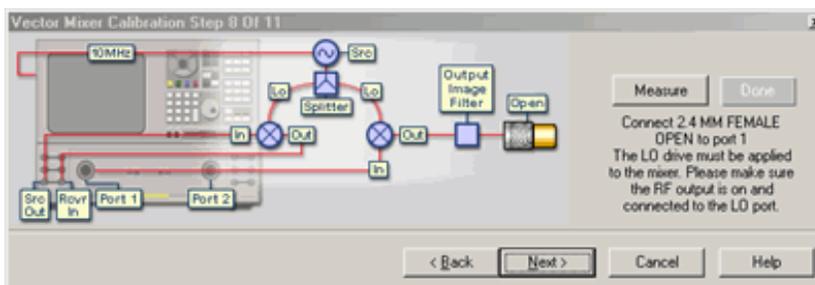
[ReMeasure] Replaces Measure after standard has been measured. Allows you to remeasure a standard.

Done Click to proceed to the [Calibration Complete](#) dialog. Available only after all measurements for the calibration are complete.

Back Returns to the previous dialog box.

Next Does NOT make a measurement. Proceeds to the next required step.

Cancel Exits the Calibration Wizard.



Vector Mixer Cal Steps dialog box help

Connect the Open, Short, and Load standards to the image filter output, then click **Measure.**'

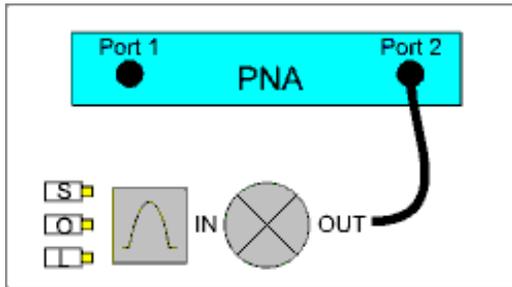
This portion of the calibration characterizes the calibration mixer.

The connection is different depending on if the calibration mixer is an upconverter being characterized as a down converter.

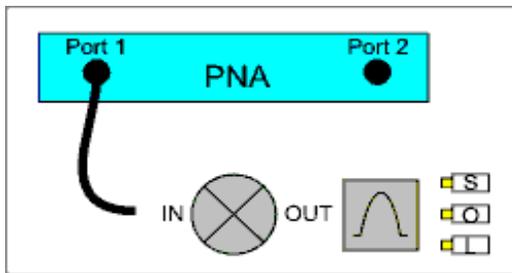
Note:

The following are **simplified** connection diagrams - the reference mixer and LO signals must also be connected. These images assume that the DUT output is connected to PNA port 2.

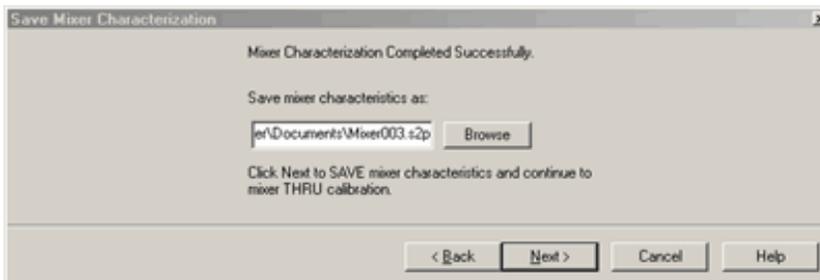
As a **Downconverter**. (The PNA automatically switches to make the S22 measurement on the device.)



As an **Upconverter**



Done Click to proceed to the [Calibration Complete](#) dialog. Available only after all measurements for the calibration are complete.



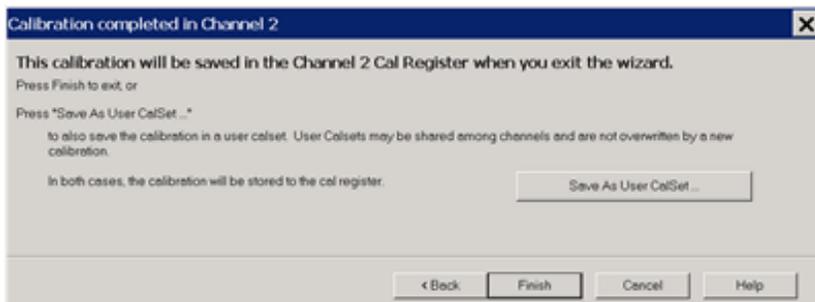
Save Mixer Characterization dialog box help

Allows you to save the characterization data of your [calibration mixer](#). When performing another VMC calibration using the same calibration mixer, this [S2P file can then be recalled](#).

Browse Navigate to the location where you want to save the characterization data of your calibration mixer. Either use the default file name or enter a custom file name.

Next Saves the mixer characterization file and continues with the next step in the full system calibration routine.

Finish Replaces **Next** if you are only characterizing the calibration mixer instead of performing a full system calibration. Saves the mixer characterization file and exits the mixer characterization routine.



Calibration Completed dialog box help

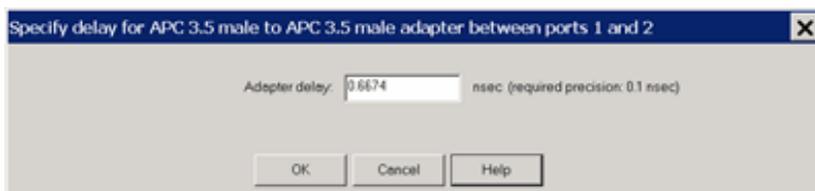
Finish Save to the channel's calibration register.

Save As User Cal Set Starts the [Save as User Cal Set dialog box](#) AND save to the channel's calibration register.

Cancel Calibration is NOT applied or saved.

Learn about [Calibration Registers](#).

Learn about [User Cal Sets](#)



Specify delay dialog box help

This dialog appears ONLY when [Adapter Removal](#) or [Unknown Thru](#) calibrations are performed.

The following values were estimated from the measurement. Most of the time, they are adequate. However, for CW sweep or frequency sweep with large step sizes, the accuracy of the values may be improved.

Adapter delay To improve this value, measure and record the delay of the adapter with a dense step size. Enter that value here. The required precision value is the accuracy that is required to characterize the delay value.

Nominal phase offset (Waveguide ONLY). To improve this value, measure and record the phase offset of the Waveguide adapter with dense step size. Enter that value here.

When one connector is coax and the other connector is waveguide, the phase offset has an ambiguity of 180 degrees. For consistency, the estimate provided here is always between 0 and 180 degrees. You can change this estimate to any value between -180 degrees and +180 degrees.

For FCA calibrations, this dialog box appears twice: once for the input frequencies and once for the output frequencies. The values can be slightly different.

Last Modified:

4-Jan-2012	Fixed LO Cal enable
21-Oct-2010	MX New topic

How to make a VMC Fixed Output Measurement

The following is a step-by-step example illustrating how to measure a mixer in swept LO mode using FCA Vector Mixer Calibration.

There are fewer components required for [SMC](#) as compared to VMC, and fewer measurement steps. You can now make relative phase measurements with SMC. Also, **ONLY** SMC (not VMC) can measure the reverse conversion loss of the mixer.

This procedure can also be used for making **fixed** LO measurements, which is quite similar. Although the LO source is still required, the physical triggering cables that connect the PNA and External Source are not required.

Required Equipment

- PNA-X or PNA 'C' models
 - with option 083 (FCA)
 - with PNA Rev A.09.33 or above

Note: This example has been updated to show keystrokes for the FCA User Interface available with PNA Rev. A.09.33 and above.

- GPIB External Source (Agilent ESG or PSG works best). This source is NOT necessary when using PNA-X with Internal Second source.
- Reference Mixer ([see requirements](#))
- Calibration Mixer/Filter ([see requirements](#))
- Power splitter - Not necessary when using PNA-X with Internal Second source.
- ECal module with connectors that match the Input and Output connectors of the DUT. You can use adapters to make the ECal module match the DUT connectors, but first perform an [ECal user-characterization](#) with the adapters attached. ECal makes the FCA calibration much easier.
- Cables and adapters
- **Optional** GPIB Power meter and sensor (for LO power calibration)

The example mixer

The example device is a mixer with the following characteristics:

- LO and Input Frequency Range: 2 GHz to 4.2 GHz

- Output Frequency Range: DC to 1.3 GHz

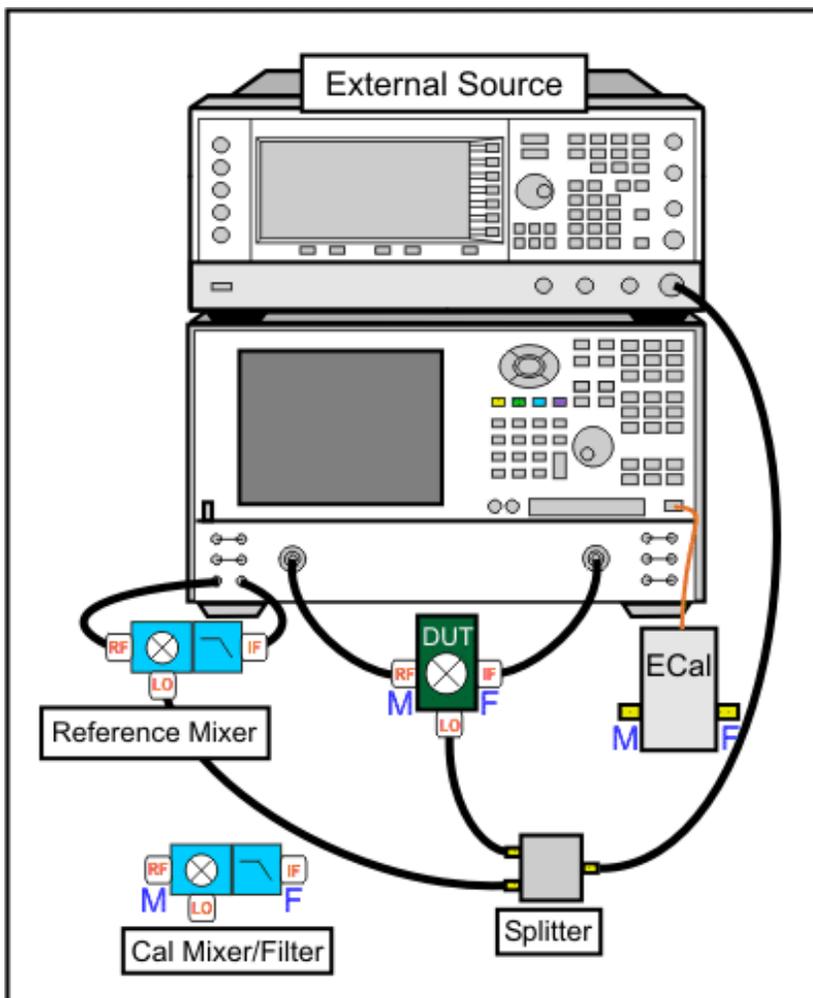
We will measure:

- Fwd Conversion Loss (VC21)
- Input match (S11)
- Output match (S22)
- Rev Conversion Loss is NOT possible because of the reference mixer.

VMC Setup

Connect the devices as shown in the following diagram:

Note: This setup can also be used for SMC measurements, allowing you to make VMC and SMC measurements simultaneously on separate channels. The Reference Mixer is automatically switched during SMC measurements. The Cal Mixer/Filter is not used.

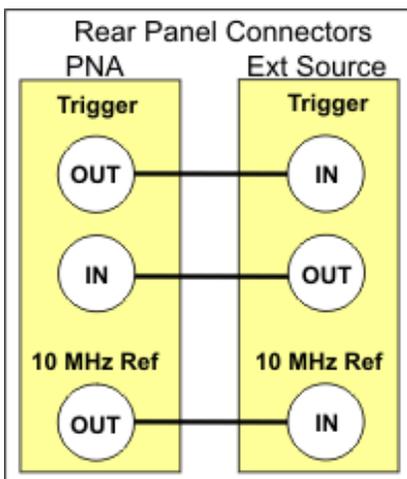


Notes:

- When using a PNA-X with an Internal Second Source, the external source is NOT necessary.
 - [See note regarding LO power out both second source ports](#)
 - [Learn which PNA ports can be used for the LO.](#)
- The low-pass filter on the output of the Reference Mixer is recommended, but NOT required. [Learn more.](#)
- When using 4-port models, the mixer input must be connected to PNA port 1. The mixer output can be connected to any other PNA port.

Make Connections on the Instrument rear panels:

1. Connect the Source to the PNA GPIB Controller port using a GPIB cable.
2. Using two BNC cables, connect the Source and PNA Trigger connectors as shown in the following image. This is not necessary when making fixed LO measurements, or using a PNA-X with Internal Second Source.
3. Using a BNC cable, connect the PNA **10 MHz Reference Output** to the Ext. Source **10 MHz Reference Input**.



Create the Measurement

For this document:

- Front-panel hardkeys are formatted as "Press **TRACE**"
- Front-panel softkeys are formatted as "Press [**S11**]"
- Menus are formatted as "Click **System**"

1. Connect the DUT.
2. Press **PRESET** to make sure you are starting with a known state.
3. Press **MEAS** then [**Measurement Class**], then **Vector Mixer/Converter** then **OK**. At the **Confirm...** dialog, click **OK**. An S11 trace is created.
4. Press [**VC21**] to replace the S11 trace.

Configure the Mixer settings

1. Press **FREQ** then [**Mixer Setup**] to start the Mixer Setup dialog.
2. On the **Sweep** tab, no changes from the default settings are required. The Avoid Spurs feature is useful for eliminating spurs in test setups with excessive LO leakage.
3. On the **Power** tab, change the DUT Input Port Power Level to -17 dBm.
4. On the **Mixer Frequency** tab, enter the frequency values as shown in the following image:

Input	Start/Stop	2.57950000 GHz	3.58050000 GHz	Calc Input
LO1	Start/Stop	1.77950000 GHz	2.78050000 GHz	<input checked="" type="checkbox"/> Input > LO
Output	Fixed	<input type="radio"/> +	4.35900000 GHz	Calc LO
		<input checked="" type="radio"/> -	800.000000 MHz	Calc Output

- a. You can enter the Input and the Output frequencies, then click **Calc LO**.
- b. If **Input > LO** is NOT checked, the PNA assumes you want the Input < LO frequencies, and higher LO frequencies are calculated as a result.

External Source Configuration

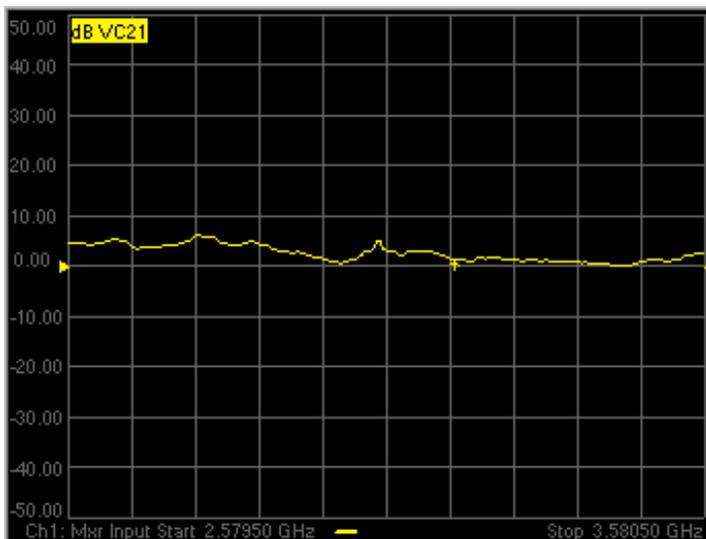
When using a PNA-X with Internal Second Source, the external source is NOT necessary.

- [See note regarding LO power out both source ports](#)
- [Learn which PNA ports can be used for the LO.](#)

1. On the **Mixer Setup tab**, click **Add Source** to start the External Source Configuration dialog. [Learn how to configure an external source.](#)
2. On the External Source Configuration dialog, click **Device Properties**.
 - a. Click **Hardware List (BNC)**, which is the fastest measurement method. This method requires the BNC Trigger cables that connect the PNA and source. If not available, **Software CW** can be used, but measurements are much slower.
 - b. If necessary, select the Interface (usually **GPIB**), then **Refresh**, then select the **Available IO Configuration**.
 - c. Click **OK** to close the Device Properties dialog, then **OK** to close the External Source Configuration dialog.
3. Next to LO1, select the configured external source name.

Complete Mixer Setup

1. On the **Mixer Power tab**, change LO1 Power to 0.00 dBm This setting specifies the power out of the external source (not at the DUT) unless an LO power cal is performed.
2. When the settings are valid, the **Save**, **Apply** and **OK** buttons are available to click. To save the mixer settings in a file so you can recall them easily, click **Save...**, then type a descriptive filename. Then click **OK**.
3. Click **OK** to close the VMC Setup dialog.
4. To change the X-axis values from the default Output selection (800 MHz), click **Response**, then **Measure**, then Input.
5. The trace is updated as the external source steps in frequency. It should look something like the following image. Because of the reference mixer, the uncorrected VMC measurement can look like it has gain.



Problems?

Not sweeping:

- On the PNA, press **TRIGGER**, then **Continuous** to start the PNA sweeping. Watch for error messages on the PNA and source.

Problems communicating with the source:

- Press **SYSTEM** then **Configure** then **External Device**. On the External Device Configuration dialog, select the external source, then click Device Properties. Next to Trigger Mode: select **Software CW trigger**, then close the dialog. Again, press **TRIGGER**, then **Continuous** to start the PNA sweeping. If this works, then something is wrong with **Hardware (BNC)**. Check the trigger cables on the rear panel.
- As a last resort, try rebooting the PNA. First, [save the entire setup to a .csa file](#). When the PNA preset measurement appears, recall the .csa file to resume at this step.

If the source is sweeping, and the PNA Input is sweeping, but there is still no output.

- Check power levels at the LO and Input.
- Check the DUT by making a fixed LO measurement which is much easier.

Perform a VMC Calibration

1. Disconnect the DUT.
2. Connect the ECal module to a PNA USB port.
3. Press **CAL**, then **[Start Cal]**, then **[Cal Wizard]**. Because the VC21 measurement is active, the Cal Wizard automatically begins a VMC Calibration.
4. At the **Calibration Setup** dialog, click **Next**. Or check **Enable LO1 Power Cal** to perform a [Source Power Cal](#) to specify the LO Power at the DUT. This requires a power meter or USB power sensor be connected.
5. At the **Calibration Mixer Characterization** dialog, click **Next**. We will perform characterization of the Calibration mixer as part of the VMC cal. Later we will save the Calibration mixer characterization so that, in future VMC calibrations that use this same frequency range, we can recall the Calibration mixer characterization by clicking **Load Characterization from file**.
6. At the **Select DUT Connectors and Cal Kits** dialog, for **Port 1** select the connector type and gender of your DUT INPUT. For **Port 2** select the connector type and gender of your DUT OUTPUT. Then select ECal as the Cal Kit to use for each connector. Click **Next**.
7. At the **Select the ECal Port to be Connected** dialog, ensure that **Port A** is selected for **Port 1**, then click **Next**.

8. At the **Vector Mixer Calibration Step 1 of 3** dialog, connect the ECal module Port A to the Port 1 cable, and Port B to the Port 2 cable. Then click **Measure**. This portion of the calibration gathers the linear (non-frequency-translating) error terms of the test setup at the input and output frequencies.
9. At the **Vector Mixer Calibration Step 2 of 3** dialog, connect the following, then click **Measure**. This portion of the calibration will connect reflection standards to characterize the S-parameters of the calibration mixer/filter.
 - Port 1 cable to the Input of the calibration mixer.
 - LO cable to the LO port of the calibration mixer.
 - ECal module to the Output of the calibration mixer/filter.
10. At the **Vector Mixer Calibration Step 3 of 3** dialog, disconnect the ECal module and connect the Port 2 cable to the output of the calibration mixer/filter, then click **Measure**. This step completes the calibration using the characterized mixer/filter as a Thru standard.
11. At the **Save Mixer Characterization** dialog, click **Browse**, then type a unique filename and click **OK**. Then click **Next**. This saves the Calibration Mixer characterization to an S2P file. This file can be recalled for subsequent VMC calibrations.
12. At the **Calibration completed** dialog, you can choose to save the VMC calibration as a User Cal Set. Otherwise, click **Finish** to complete the VMC calibration. Correction is turned ON and applied to the VMC trace that we set up earlier.

What is happening?

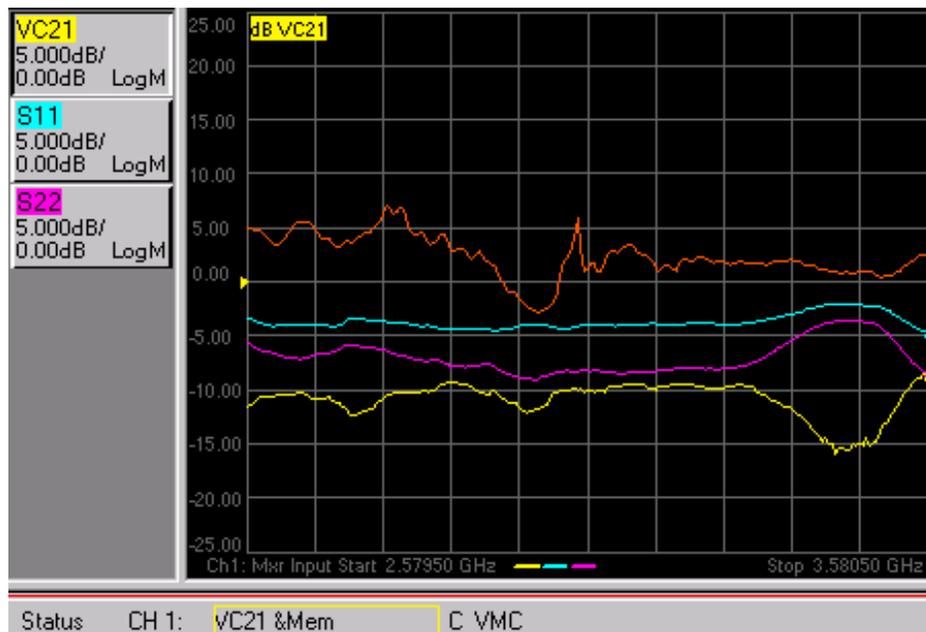
When an external source is sweeping, the measurements are much slower. When correction is ON, you will see that there are times when nothing is happening on the screen. This is because there are background measurements being made but not displayed.

This is exactly the same as when full 2-port correction is applied to an S-parameter. All four parameters are measured, then correction is applied, then all four measurements are updated. This occurs much faster when there is no external source. With a VMC measurement, there is no VC12 (reverse transmission measurement), so there are only three background measurements. With correction OFF, the traces are updated as the data is measured. You can see this taking place by creating the following measurements.

Create S11 Input and S22 Output Match

1. Press **MEAS** then [**S11**] and [**S22**] to add these measurements to the same channel.
2. While the source is sweeping, watch the source port indicator on the front of the PNA. First, the port 1 indicator will light for two sweeps, then the port 2 indicator will light for 1 sweep while all 3 traces update.
3. Press **CAL**, then [**Correction OFF**]. Notice that the relevant traces will update as the sweep is occurring.

The following image shows the corrected Conversion Loss (VC21), Input Match (S11), Output Match (S22) and the uncorrected Conversion Loss (VC21), which is a memory trace.



Last Modified:

- | | |
|-------------|-------------------------------|
| 4-Apr-2011 | Updated for FCA2 (A.09.33) |
| 16-Apr-2009 | Updated for PNA-X |
| 6-Apr-2009 | Replaced N5242a with PNA-X |
| 5-May-2008 | Added selectable output note. |
| 17-Apr-2008 | Added 10 MHz Ref image |

Embedded LO Measurements

The Embedded LO feature allows you to make [VMC](#), [SMC](#), [IMDx](#), [GCX](#), and [NFX](#) measurements of mixers that have a FIXED LO inside the DUT. Learn how to make [IMX Spectrum](#) measurements on converters with an Embedded LO.

Note: This feature is available as Opt 084, and must be [enabled](#).

In this topic:

- [Overview - How the PNA measures the embedded LO](#)
- [To measure a DUT with an Embedded LO \(Procedure\)](#)
- [How to Launch the Embedded LO Mode dialog box](#)
- [Embedded LO dialog box help](#)
- [Embedded LO Diagnostic dialog box help](#)

Overview - How the PNA measures the embedded LO

Measurements of these devices are challenging for a couple of reasons:

1. The VMC measurement process requires the use of a [reference mixer](#) that has the same LO frequency as the DUT. A separate internal or external source must be used for the reference mixer LO. A PNA with an internal second source is much faster. **This source (Internal or External) MUST be controlled by the PNA!** [Learn how.](#)
2. All Embedded LO measurements require the PNA receivers to be tuned to the correct frequency to measure the mixer output, which is highly dependent on the exact LO frequency.

The nominal frequency of the embedded LO is input into the [Mixer Setup dialog](#). This is used as a starting point for the measurement.

Before each DUT measurement sweep, background sweeps are made to determine the frequency of the embedded LO to a configurable degree of accuracy.

Background sweeps...

- **Broadband Sweep** - rough measurement of the embedded LO frequency, made around a selectable data point over a selectable frequency span. The input signal to the DUT is tuned to a selectable CW frequency. The B receiver is swept across a selectable span around the anticipated output frequency. The difference between the frequency of the found signal and the desired output frequency is then applied as an adjustment.
- **Precise Sweep** The B receiver is measured at the selectable data point. Measurements of phase versus time are made, from which the exact offset frequency is computed, until either the tolerance value or

maximum iterations are met.

- For VMC measurements, the reference mixer frequency is updated as the embedded LO frequency is determined.

To measure a DUT with an Embedded LO:

1. Create a [VMC](#), [SMC](#), [IMDx](#), [IMx Spectrum](#), [GCX](#), or [NFX](#) measurement.
2. In the mixer setup dialog, enter the nominal frequency of the embedded LO as the LO frequency.
3. Perform a calibration as usual.
4. [Launch and complete the Embedded LO Mode dialog box](#) (below)

For VMC with Embedded LO:

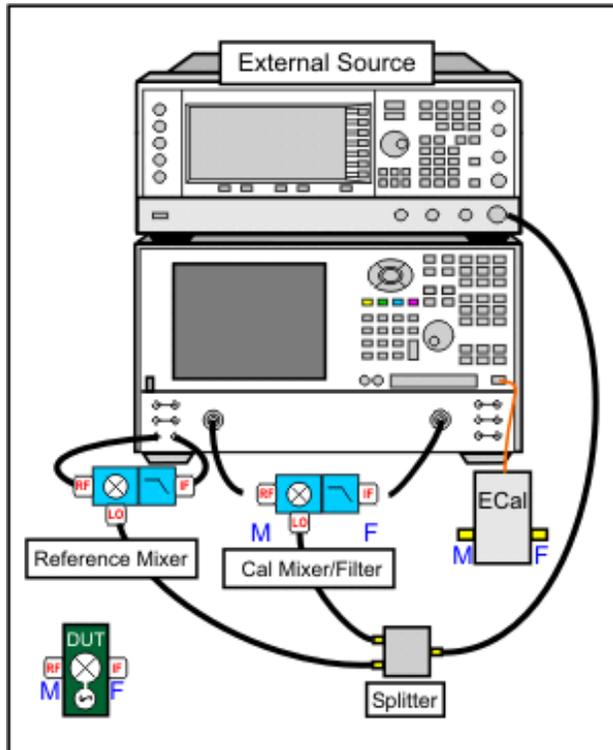
(The other mixer applications do NOT require a reference mixer.)

The LO source for the Reference Mixer can be either:

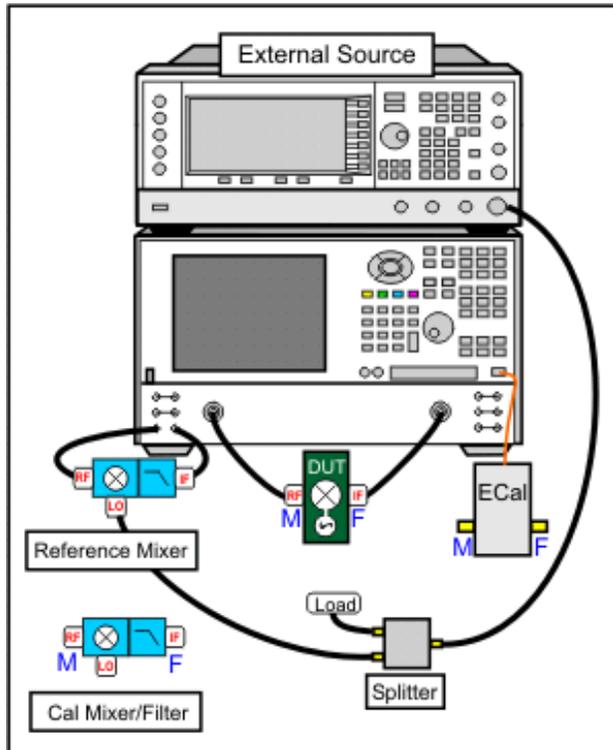
- An **Internal** source when using a [PNA-X that has two sources](#).
- An **External** source:
 - Must be configured as an External Device. [Learn how](#).
 - Must be locked to the PNA using the [10 MHz reference](#).

This source (Internal or External) MUST be controlled by the PNA. To control the source, set the LO1 source in the [Mixer Setup tab](#) to whichever source (external, port 3, or port 4) that is driving the reference mixer.

During Calibration - The LO source is shared between the Reference Mixer and the Calibration Mixer/Filter. This requires a splitter when using an external source, as shown in the following image.



During the Measurement - Only the Reference Mixer uses the LO source. Terminate the LO source port that is no longer used by the Calibration Mixer/Filter to ensure that the match seen by the Reference Mixer LO port does not change after the calibration, as shown in the following image. This precaution is not necessary when using the internal second source (ports 3 and 4) of the PNA-X.



For SMC, IMDx, IMxSpectrum, and NFX measurements:

No unique setup is required for embedded LO measurements.

How to Launch the Embedded LO Mode dialog box

Using front-panel HARDKEY [softkey] buttons

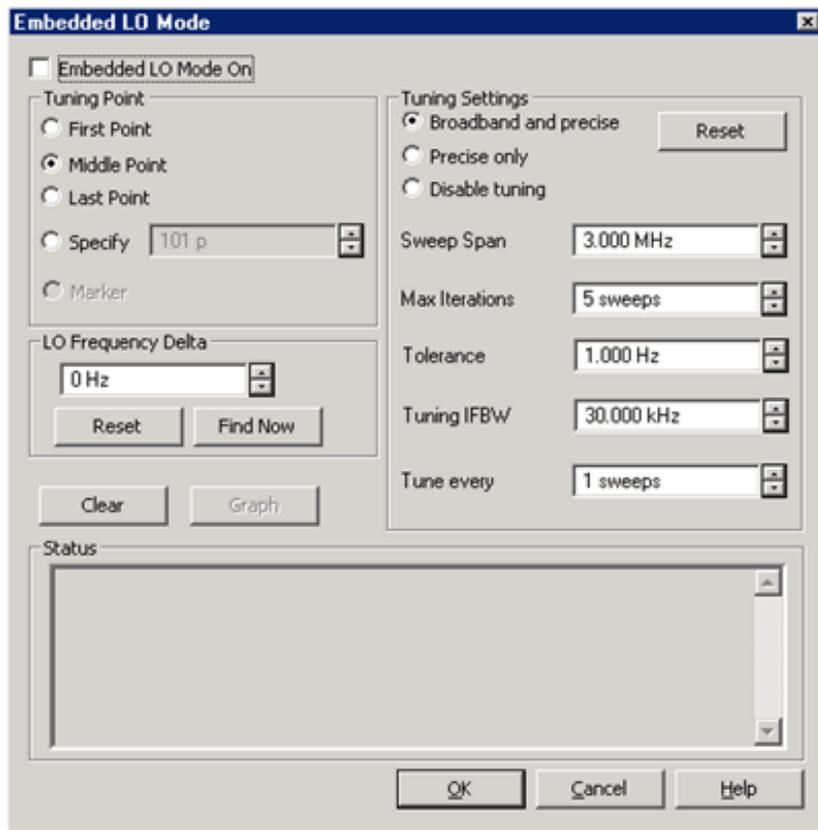
1. Not Available

Using a mouse with PNA Menus

1. Click **Response**
2. then **Measure**
3. then **Embedded LO**

Programming Commands

Embedded LO dialog box help



The Tuning Settings balance LO measurement speed versus accuracy. You can see that accuracy is becoming compromised when noise starts to appear on the measurement trace.

[Scroll up](#) to learn more about the Embedded LO measurement process.

Embedded LO Mode On Check to enable measurement of the Embedded LO.

Tuning Point Select, or specify, the data point in the mixer sweep that will be used to find the embedded LO frequency. If a marker is enabled, that data point can be used. For broadband and Precise sweeps, choose a point in the mixer sweep where noise is least likely to be found, such as the point of highest gain. This is generally the center of a sweep or the center of a filter if used.

LO Frequency Delta The absolute difference between the measured embedded LO frequency and the LO setting that is entered in the [Mixer Setup dialog](#). This value is updated each time the embedded LO frequency is measured. Entering a value is a way to change the LO frequency on the mixer setup without invalidating the calibration.

Reset Set the LO Frequency Delta back to 0 Hz.

Find Now The PNA finds and measures the actual LO frequency using the current dialog settings. This data is displayed in the **Status** box.

Tuning Settings These settings determine the amount of time spent versus the degree of accuracy

to which the LO Frequency is measured. You can see that accuracy is becoming compromised when noise starts to appear on the measurement trace.

Reset Set all Tuning Settings back to the defaults.

Broadband and Precise Does the entire tuning process for each background sweep. See the [Overview](#) for more information.

Precise only Does NOT perform broadband tuning on each sweep. Use this setting when the embedded LO is stable. The signal (after broadband) must be within $\frac{1}{2}$ the tuning IFBW. If the signal will always be within $\frac{1}{2}$ the IFBW, broadband tuning is not needed. Most satellite components are within 3 kHz absolute so might not need broadband tuning.

Disable tuning Only the previously measured LO Frequency Delta is applied to the reference mixer LO and PNA receivers.

Sweep Span Narrowing the sweep span limits the number of data points that are measured in the broadband sweep and makes the measurement faster. Sweep span should be less than 14 MHz. The resolution of the broadband mode is the tuning IFBW.

Max Iterations The maximum number of Precise sweeps to make. When this number is reached, the final measurement is used.

Tolerance When two consecutive Precise measurements are made within this value, the final measurement is used. If this is not achieved within the Max Iterations value, then the last measurement is used. This is the best of the 'Tunings settings' to change to improve accuracy.

Tuning IFBW IF Bandwidth used for Broadband and Precise tuning sweeps. This sets the resolution in the Broadband sweeps and sets the max error ($\frac{1}{2}$ IFBW) for precise tuning. The larger the IFBW, the faster the sweep, but the signal may not be found.

Tune every Set the interval at which tuning is performed before a measurement sweep. 'Tune every 3 sweeps' means that every third measurement sweep is preceded by tuning sweeps. If the embedded LO drifts, or if regularly changing DUTs, use 'Tune every 1 sweep'.

Status Allows textual and graphical representation of the Embedded LO measurement sweeps.

Clear Removes the text information currently being displayed.

Graph Launches the following graphical (spectrum analyzer type) display sweeps of the latest embedded LO measurement.

Frequency Offset Mode

Frequency Offset Mode (FOM) provides the capability to have the PNA Sources tune to frequencies that are different (offset) from the PNA Receivers.

PNA Option 080 provides you with the hardware and basic software capability to make Frequency Offset Measurements. This topic discusses the PNA settings that are relevant to making these types of measurements. See [Frequency Converting Device Measurements](#) for more information on making specific device measurements.

- [Frequency Offset Dialog Box](#)
- [Setup Examples](#)
- [Test Set \(Reference Switch\) Dialog Box](#)

Other Frequency Offset topics

How to make Frequency Offset settings

Using front-panel HARDKEY [softkey] buttons

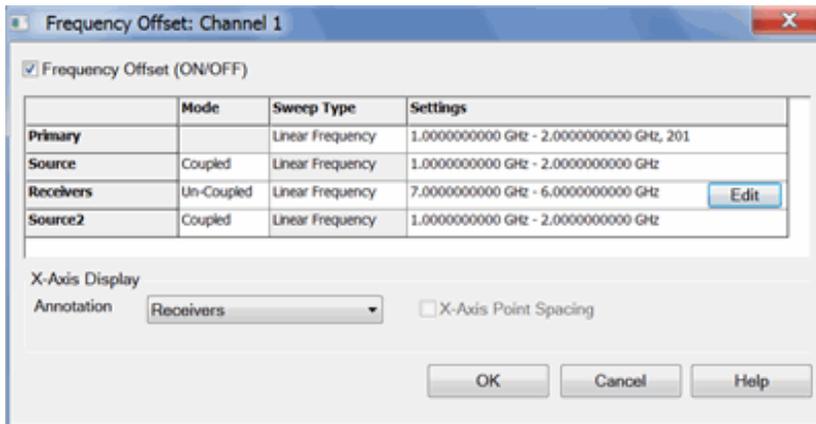
1. Press **STIMULUS**
2. then **[Frequency Offset]**

PNA Menu using a mouse

1. Click **Stimulus**
2. then **Frequency**
3. then **Frequency Offset**

[Programming Commands](#)

Frequency Offset dialog box help



The following are major changes to FOM:

- Stimulus and Response are now called Sources and Receivers.
- Sources and Receivers settings can be made in two ways:
 1. By **Coupling** to the Primary (Channel) settings. This is the only method used in previous releases.
 2. By **Uncoupling** and setting Sources and Receivers values independently. This is the new, simplified method.
- External sources appear here and can be controlled from this dialog. [Learn more.](#)

Note: Source2 supplies power for ports 3 and 4. **Turn Source2 power ON** using the [Power and Attenuators dialog](#). This (Frequency Offset) is the only dialog for controlling the frequency of Source 2. Learn more about [Source2](#).

Frequency Offset (ON/OFF) Enables Frequency Offset Mode on ALL measurements that are present in the active channel.

When FOM is NOT enabled, all frequencies are the same as the active channel.

Tip: First make other settings on this dialog box, then click **Frequency Offset ON**.

Primary The current Active Channel settings. When a Source or Receiver is coupled to the Primary settings, its Sweep Type is the same as that of the Primary. The frequency settings of the coupled range are mathematically derived from the Primary settings using the [Multiplier, Divisor, and Offset values](#). With this approach, only the Primary settings need to be changed in order to affect change in the coupled Sources and Receivers. Changes to the Primary channel settings occur when Frequency Offset is checked ON. [See example using Primary and Coupled setting](#).

Tip: Primary settings are ONLY used when Sources and Receivers are Coupled. It is often easier

to Uncouple, then set Sources and Receivers independently.

Source and Source2 if available. [Learn more about Internal Second Source.](#)

Receivers All receivers that are used in the channel, including Reference receivers, are tuned to the specified frequency settings.

Mode

Coupled Source and Receiver settings are mathematically derived from the Primary settings using Multiplier, Divisor, and Offset values. [Learn more.](#)

Uncoupled Source and Receiver settings are entered independently, without reference to Primary settings. When Uncoupled, Source and Receiver Ranges can use separate sweep types.

Sweep Type Click to change the type of sweep for each range. Only available for Primary and Uncoupled Sources and Receivers.

Unsupported Sweep Type combinations

- Power Sweep and Segment Sweep can NOT be used together.
- Uncoupled Log Sweep yields **invalid data** whenever the sources are offset from the receivers.
- Coupled Log Sweep is allowed only for the following two conditions:
 1. The offset = 0, the multiplier = 1, and the divisor = 1.
 2. The multiplier = 0

Settings To change settings, click **IN** the appropriate Settings cell, then click **Edit**.

- If coupled, invokes the [Coupled dialog](#).
- If uncoupled or Primary invokes the [Uncoupled settings dialog](#).

X-Axis Select the settings to be displayed on the X-Axis.

X-Axis Point Spacing Only available when a Segment Sweep Type is selected as the X-Axis display. [Learn more.](#)

Note: When Frequency Offset is enabled, ALL receivers on the channel, including the reference receivers, tune to the new offset frequencies, Therefore the source and reference receiver will be at different frequencies. Therefore, FOM measurements that include a reference receiver, which includes all S-parameters, display invalid data.

To measure and display measurements at both the source and receiver frequencies, you must use two channels. Use [Equation Editor](#) to calculate the conversion loss. [See a calibrated FOM conversion loss example.](#)

[Learn how to calibrate frequency offset measurements.](#)

Coupled settings dialog box help



Coupled Formulas:

Range Start = [Primary Start x (Multiplier / Divisor)] + Offset

Range Stop = [Primary Stop x (Multiplier / Divisor)] + Offset

Where:

Offset Specifies an absolute offset frequency in Hz. For mixer measurements, this would be the LO frequency. Range is +/- 1000 GHz. Offsets can be positive or negative.

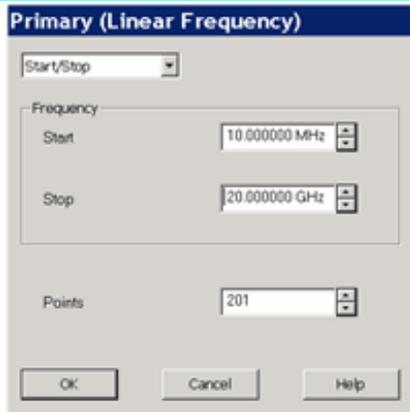
Multiplier Specifies (along with the divisor) the value to multiply by the stimulus. Range is +/- 1000.

- Negative multipliers cause the stimulus to sweep in decreasing direction. For downconverter mixer measurements, this would be for setups requiring the Input frequency to be less than LO frequency. [See an example.](#)

- 0 (zero) as the multiplier nulls the Primary setting. Then the Offset value adds to zero.

Divisor Specifies (along with the multiplier) the value to multiply the stimulus. Range is 1 to 1000.

Primary and Uncoupled settings dialog box help



This dialog will vary depending on the sweep type:

Linear and Log frequency

Uncoupled Log sweep yields **invalid data** whenever the sources are offset from the receivers.

Select Start/Stop or Center/Span

Frequency Enter values

Points (Primary only) Enter number of data points for the sweep.

Power

CW Freq Enter frequency in Hz.

Points (Primary only) Enter number of data points for the power sweep.

CW Time

CW Freq Enter frequency in Hz.

Sweep Time Enter time to complete one sweep. Enter 0 for the fastest sweep.

Segment Sweep Edits are made exactly like the [standard segment table](#).

For Advanced Users: Uncoupled Segment Sweep offers great flexibility in configuring

measurements. In segment sweep mode:

- The **OK** button is NOT available until the total number of data points for all segments matches the number of Primary data points.
- [Independent IF Bandwidth](#) and [Independent Sweep Time](#) are available ONLY on the Primary (channel) and the Uncoupled **Receivers** - NOT Sources.
- [Independent Power](#) is available ONLY on the Primary (channel) and the Uncoupled **Sources** - NOT Receivers.

Setup Examples

Although the Frequency Offset settings can be used with many types of devices, these examples include mixer terminology.

[See a Mixer Compression and Phase \(AM-PM\) Measurement using FOM.](#)

[See a calibrated FOM conversion loss example.](#)

1. Fixed LO - Upconverter

- **Swept Stimulus (Mixer Input):** 1000 MHz - 1200 MHz
- **Fixed LO:** 1500
- **Swept Response (Mixer Output):** 2500 MHz to 2700 MHz

Make the following settings on the FOM dialog

Source: Uncoupled

Sweep Type: Linear

Click Settings, then Edit. In the Source dialog:

Start Frequency = 1000 MHz

Stop Frequency = 1200 MHz

Receiver: Uncoupled

Sweep Type: Linear

Click Settings, then Edit. In the Receiver dialog:

Start Frequency = 2500 MHz

Stop Frequency = 2700 MHz

LO Settings

Set external source to CW - 1500 MHz.

Source2: Uncoupled (Only with [Second PNA Internal Source](#))

Sweep Type: CW Time

Click Settings, then Edit. In the Source2 dialog:

CW Frequency = 1500 MHz

2. Fixed LO - Downconverter (Input < LO)

- **Swept INCREASING Stimulus (Mixer Input):** 1000 MHz to 1100 MHz
- **Fixed LO:** 2500 MHz
- **Swept DECREASING Response (Mixer Output)** 1500 MHz to 1400 MHz

Make the following settings on the FOM dialog

Primary: Not used

Source (Input): Uncoupled

Sweep Type: Linear

Click Settings, then Edit. In the Source dialog:

Start Frequency = 1000 MHz

Stop Frequency = 1100 MHz

Receiver (Output): Coupled

Sweep Type: Linear

Click Settings, then Edit. In the Receiver dialog:

Offset: 2500 MHz

Multiplier: -1 (Minus one)

LO Settings

- If using external source, set to CW: 2500 MHz.
- If using **Source2**: Set to Uncoupled, then:

Sweep Type: CW Time

Click **Settings**, then **Edit**. In the Source2 dialog:

CW Frequency = 2500 MHz

[See a calibrated FOM conversion loss example.](#)

3. Swept LO - Fixed Output - Upconverter

Swept External LO measurements in Frequency Offset Mode can be very difficult. The external LO source must be synchronized with the swept output or input (as in this case). See [Synchronizing and External Source Control](#) to see how this is done. The [Frequency Converter Application Opt 083](#) performs makes these measurements easily.

- **Swept Stimulus (Mixer Input)**: 1000 MHz to 1100 MHz
- **Swept LO**: 1500 MHz to 1400 MHz
- **Fixed Response (Mixer Output)**: 2500 MHz

Make the following settings on the FOM dialog

Source: Uncoupled

Sweep Type: Linear

Click Settings, then Edit. In the Source dialog:

Start Frequency = 1000 MHz

Stop Frequency = 1100 MHz

Receiver: Uncoupled

Sweep Type: CW Time

Click Settings, then Edit. In the Receiver dialog:

CW Frequency = 2500 MHz

LO Settings

- If using external source, set to sweep from 1500 - 1400 MHz.
- If using **Source2** ([Second Internal Source](#)): set to Uncoupled, then:

Sweep Type: Linear

Click Settings, then Edit. In the Source2 dialog:

Start Frequency = 1500 MHz

Stop Frequency = 1400 MHz

4. Power Sweep for Mixers

To measure the gain compression of a mixer, the input power to the mixer is swept. The input and output frequencies are fixed but offset from one another.

This is a good use of Coupled settings because the same compression test can be performed at several different frequencies. With coupled Source and Receiver ranges, the Primary (channel) frequency can be easily changed from the front panel. The coupled source and receiver frequencies will update accordingly.

- **Swept Input Power:** -10 dBm to 0 dBm
- **Fixed Input Frequency:** 1500 MHz
- **Fixed LO:** 500 MHz
- **Fixed Output:** 2000 MHz

Make the following settings on the FOM dialog

Primary:

Sweep Type: Power Sweep

Click Settings, then Edit. In the Primary dialog:

CW Frequency = 1500 MHz

Source: Coupled

Default settings make CW Frequency: 1500 MHz (same as Primary)

Receiver: Coupled

Default settings make Sweep Type: CW Time

Click Settings, then Edit. In the Receiver dialog:

Offset = 500 MHz

LO Settings

- If using external source, set to CW: 500 MHz.
- If using **Source2** ([Internal Second Source](#)), set to Coupled, then:

Sweep Type: Power Sweep

Click Settings, then Edit. In the Source2 dialog:

CW Frequency = 500 MHz

Test Set Reference Switch

PNA-X and N522x models have a switch in the test set that allows you to bypass the port 1 reference receiver through the front panel Reference 1 connectors. This switch lets you easily switch between standard S-Parameter measurements and measurements using a reference mixer. You could use this feature to make standard S11 measurements and converter transmission measurements relative to a reference ("golden") mixer.

Note: The Frequency Converter Application [Option 083](#) simplifies the task of making extremely accurate phase measurements on MOST frequency converting devices.

How to access the Test Set dialog box

Using front-panel HARDKEY [softkey] buttons

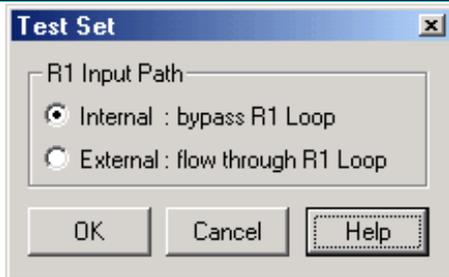
1. Press TRACE/CHAN
2. then [Channel]
3. then [Hardware Setup]
4. then [Path Config]

PNA Menu using a mouse

1. Click **Channel**
2. then **Hardware Setup**
3. then **Path Config**

◀ Programming Commands ▶

Test Set dialog box help



Note: This feature is available on PNA-X and N522x models.

R1 Input Path

Internal: bypass R1 Loop Connects the port 1 source directly to the R1 receiver.

External: flow through R1 Loop Allows direct access to the R1 receiver through the Reference 1 front-panel connectors.

See [specifications](#) which include a block diagram of reference switch.

Last modified:

- 25-Feb-2014 Minor edits
- 6-Apr-2012 Removed Opt 081
- 6-Mar-2012 Added links to calibrated example
- 30-Apr-2009 Added PNA-X models for test set switch
- 3-Sep-2008 Removed legacy content
- 25-Feb-2008 Added link to AM-PM procedure
- 16-Oct-2007 Minor edits
- 11/21/06 MQQ Modified for new dialog

Frequency Converting Device Measurements

Many frequency offset measurements can be made using the PNA with option 080. The following is a list of some of those measurements and how they are made.

- [Conversion Loss](#)
- [Conversion Compression](#)
- [Return Loss and VSWR](#)
- [Isolation](#)
- [Harmonic Distortion](#)

See Also: [Frequency Offset Measurement Accuracy](#)

Frequency Offset Measurement Accuracy

This topic discuss methods that can be used to make accurate frequency offset measurements.

- [Calibrations](#)
- [Mismatch Errors](#)
- [Accurate and Stable LO](#)

[See other Mixer Measurement topics](#)

Calibrations

With Frequency Offset measurements, the stimulus and response frequencies are different. Standard calibration error terms are calculated using reference measurements. Therefore, traditional calibration methods such as full 2-port SOLT cannot be used with frequency offset.

[Source and Receiver Power calibrations](#) can be used to calibrate your Frequency Offset measurements.

[Frequency Converter Application](#) (option 083) offers fully calibrated scalar and vector frequency offset measurements.

Source Power calibration:

- Sets accurate power level at stimulus frequencies regardless of the receiver that will be used in the measurement.
- Can be copied to other channels with copy channels feature.
- Can be interpolated.

Receiver Power Cal:

- Requires a source cal to have already been performed and applied.
- Cannot be copied to other channels.

Therefore:

- Start by performing a [source power cal](#) over the combined stimulus and response frequencies.
- [Copy the channel](#) to other needed channels and the source power cal is copied.

- Change the frequency range of the copied channel to response frequencies.
- Perform a [receiver cal](#) at the response frequencies on individual channels.
- Change the frequency range to stimulus frequency and switch [frequency offset ON](#).
- On [Status Bar](#), ensure that source and receiver cals are ON (source cal will be interpolated).

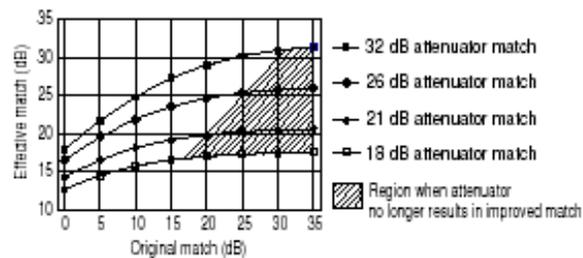
See [Frequency Offset Conversion Loss Measurements](#) to see a step-by-step example.

Mismatch Errors

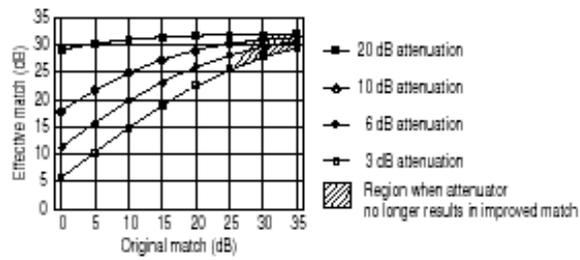
Mismatch errors result when there is a connection between two ports that have different impedances. With S-parameter measurements, these mismatches are measured and mathematically removed during a full 2-port calibration. This is much more difficult with frequency offset measurements. A much easier solution is to use high-quality attenuators on the input and output of the mixer.

By adding a high-quality attenuator to a port, the effective port match can be improved by up to twice the value of the attenuation. For example, a 10-dB attenuator, with a port match of 32 dB, can transform an original port match of 10 dB into an effective match of 25 dB. However, as the match of the attenuator approaches the match of the original source, the improvement diminishes.

Note: The Frequency Converter Application (option 083) uses calibration techniques that correct for mismatch errors.



The larger the attenuation, the more nearly the resulting match approaches that of the attenuator, as shown in the following graphic. However, excessive attenuation is not desired because that will decrease the dynamic range of the measurement system.



Accurate and Stable LO

When using frequency offset mode, if the LO signal is not accurate and stable, the output signal will not be at the expected response frequency. As a result, the output signal can fall on the skirts of the PNA receiver IF filter, or fall completely outside of the receiver filter passband.

Also, the LO power level is critical in mixer measurements. Be sure to monitor these power levels closely.

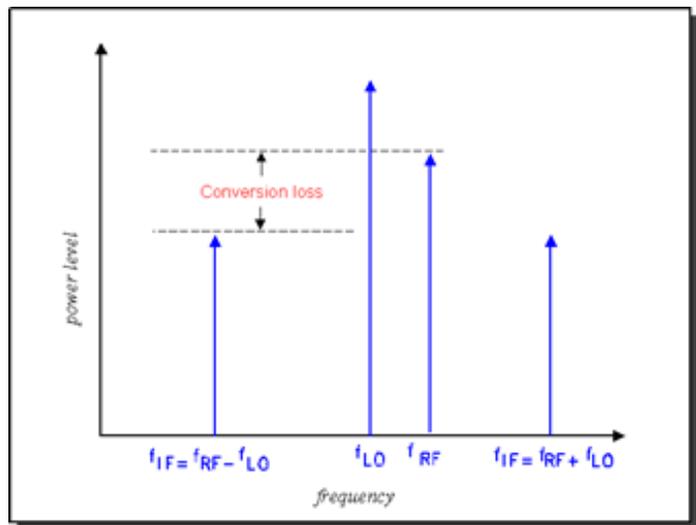
Conversion Loss (or Gain)

- [What is Conversion Loss?](#)
- [Why Measure Conversion Loss?](#)
- [How to Measure Conversion Loss](#)

[See other Frequency Converting Device Measurements](#)

What is Conversion Loss?

Conversion loss is defined as the ratio of the power at the output frequency to the power at the input frequency with a given LO (local oscillator) power. This is illustrated in the graphic below. A specified LO power is necessary because conversion loss varies with the level of the LO, as the impedance of the mixer diode changes.



Why Measure Conversion Loss?

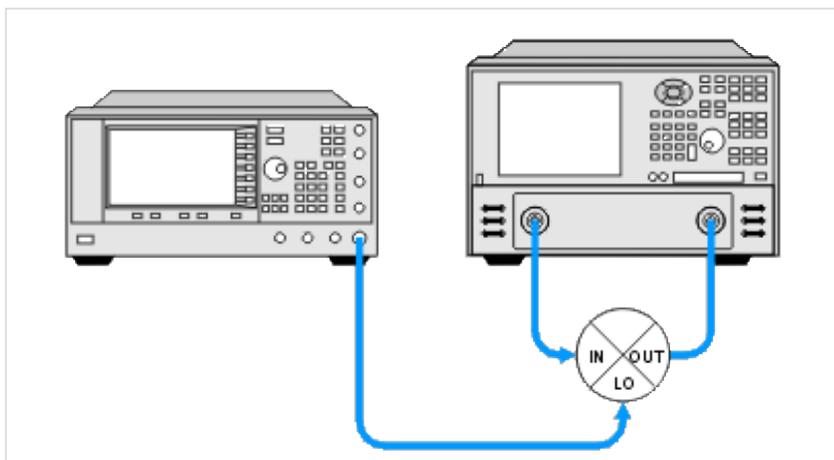
Conversion loss (or gain in the case of many converters and tuners) is a measure of how efficiently a mixer converts energy from the input frequency to the output frequency. If the conversion loss response of a mixer or converter is not flat over the frequency span of intended operation, valuable information may be lost from the resulting output signal.

How to Measure Conversion Loss

Conversion loss is a transmission measurement. It is measured by applying an input signal (stimulus) and an LO signal at specific known power levels, and measuring the resulting output signal level. Because the output frequency is different from the input frequency, [frequency offset](#) mode (option 080) must be used for this measurement.

Note: This measurement is made much easier if your PNA has the [Frequency Converter Application](#)

Equipment Setup



Example: A calibrated Conversion Loss (Down-converter) measurement

Swept Input with Fixed LO = Swept Output

- RF Input: 3.1 - 3.3 GHz
- LO: 2.2 GHz
- IF Output: 900 - 1100 MHz

PNA setup and calibrate on channel 1

1. On channel 1 create an [unratioed](#) R measurement over the ENTIRE input and output frequency span (.9 - 3.3 GHz). This will be the base source power cal that will be copied to the R and B channel measurements.
2. Perform a [source calibration](#) using a power meter. This makes the power level at the input of the mixer very accurate.

Setup Reference measurement on channel 2

1. [Copy channel](#) 1 to channel 2 which will display the reference input to the mixer. The channel 1 source power

cal is copied with the other channel settings.

2. Change measurement to R1 unratioed.
3. Change RF Input frequency to 3.1 - 3.3 GHz. The source power cal becomes interpolated.
4. Perform [receiver power cal](#). Do not need to make physical connections. The PNA source is internally connected to the R1 receiver. Makes the R receiver read the source power level.

Setup B measurement on channel 3

1. Copy channel 1 to channel 3. This channel will display the output of the mixer. The channel 1 source power cal is copied with the other channel settings.
2. Change measurement to B unratioed.
3. Change IF Output frequency to .9 - 1.1 GHz. This causes the source power cal becomes interpolated.
4. Connect thru line from port 1 to port 2.
5. Perform receiver power cal. This makes the B receiver read the source power at the IF Output frequencies.
6. [Turn OFF receiver power cal](#). This prevents an error when changing to input frequencies (next step).
7. Change RF Input frequency to 3.1 - 3.3 GHz. This changes the channel back to the mixer RF Input frequencies.
8. [Enable Frequency Offset](#).
9. Change Offset to (-2.2 GHz). This tunes the B receiver to the IF Output frequencies .9 to 1.1 GHz. **Note:** The minus sign indicates a down-converter measurement.
10. Turn ON receiver power cal.

Measure the Mixer

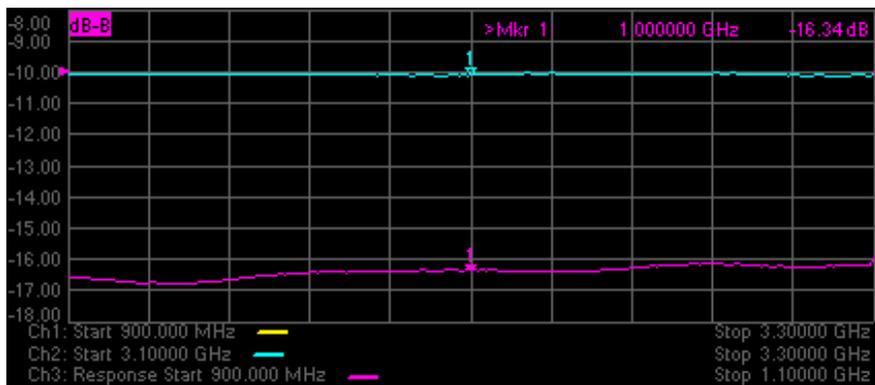
1. Connect the mixer.
2. Adjust [scale](#) to suit your needs.
3. Enable [markers](#) to read power levels for each trace.

The display below shows:

- Ch3 B receiver (bottom trace) absolute output power.
- Ch2 R1 receiver measurement (top trace) absolute input power to the mixer.

With this method, the conversion loss math (B/R1) can be performed with [Equation Editor](#) (not shown). The B/R1 ratio measurement is not supported with receiver power Cal turned on. However,

conversion loss (C21) measurements can be made directly and are much easier using the Frequency Converter Application, FCA (Opt 083).



Conversion Compression

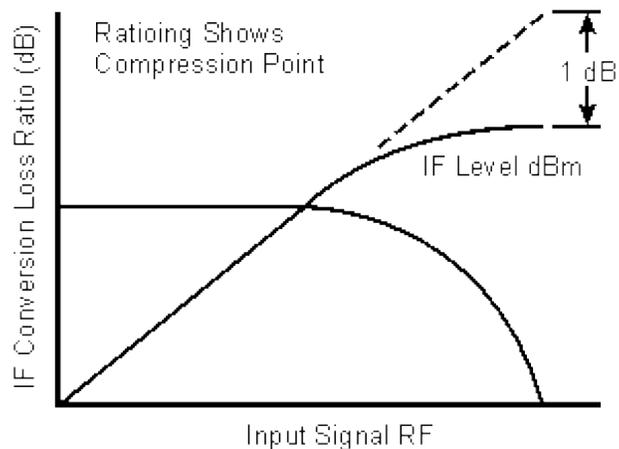
- [What is Conversion Compression?](#)
- [Why Measure Conversion Compression?](#)
- [How to Measure Conversion Compression](#)
- [Measurement Accuracy Considerations](#)

[See other Frequency Converting Device Measurements](#)

What is Conversion Compression?

Conversion compression is a measure of the maximum input signal level for which a mixer will produce linear operation. It is very similar to the [gain compression](#) experienced in amplifiers.

To understand conversion compression, you must first understand [conversion loss](#). This is the ratio of the mixer output level to the mixer input level. This value remains constant over a specified input power range. When the input power level exceeds a certain maximum level, the constant ratio between input and output power levels begins to change. The point at which the ratio has decreased 1 dB is called the 1-dB compression point. This is illustrated in the graphic below.



Why Measure Conversion Compression?

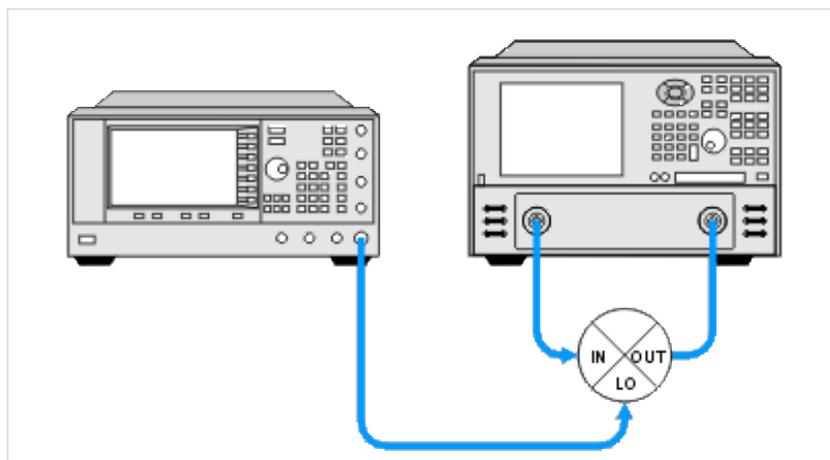
Conversion compression is an indicator of the dynamic range of a device. Dynamic range is generally

defined as the difference between the noise floor and the 1-dB compression point.

How to Measure Conversion Compression

The equipment and setup used to measure conversion compression are essentially the same as for measuring conversion loss and is illustrated in the following graphic.

The PNA performs a power sweep using [frequency-offset mode](#) and the resulting display shows the mixer's output power as a function of its input power. The 1-dB compression point (or others such as 3-dB) can be determined using markers.



Measurement Accuracy Considerations

Equipment Setup Considerations

- The couplers in the PNA have very good directivity. If the return loss of the DUT is bad, the reflected signal gets sampled by the PNA and can result in errors. This relates to error in DUT gain. To increase the accuracy, an attenuator can be added between the PNA's source port and the DUT's input port. Normally a 6- to 10-dB attenuator is sufficient. Addition of this attenuator, however, decreases the available drive to the DUT.
- With high drive levels the PNA can be driven into compression resulting in measurement error. With excessive drive levels, the PNA can be damaged. Add an attenuator between the output of the DUT and the receiver input of the PNA to avoid these problems.

Calibration Considerations

- [Source power calibration](#) can be used to provide a high level of accuracy for this measurement.

Isolation Measurements of Frequency Converting Devices

- [What is Isolation?](#)
- [Why Measure Isolation?](#)
- [How to Measure Isolation](#)

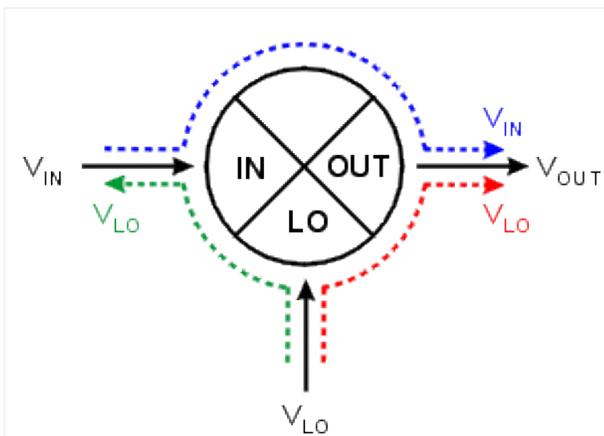
[See other Frequency Converting Device Measurements](#)

What is Isolation?

Isolation is a measure of the leakage, or feedthrough, from one port to another. The more isolation a mixer provides, the lower the amount of feedthrough. Isolation is measured at the same frequency as the stimulus, not the converted or shifted frequency. Therefore, Frequency Offset capability is not necessary for these measurements.

Three main isolation terms are of interest for mixer measurements:

- LO-to-OUT isolation (V_{LO})
- LO-to-IN isolation (V_{LO})
- IN-to-OUT feedthrough (V_{IN})



Why Measure Isolation?

Any unwanted signal "leaking" through the device will mix with the desired output signal creating intermodulation products, adding to intermodulation distortion. These unwanted signals may be difficult to filter out.

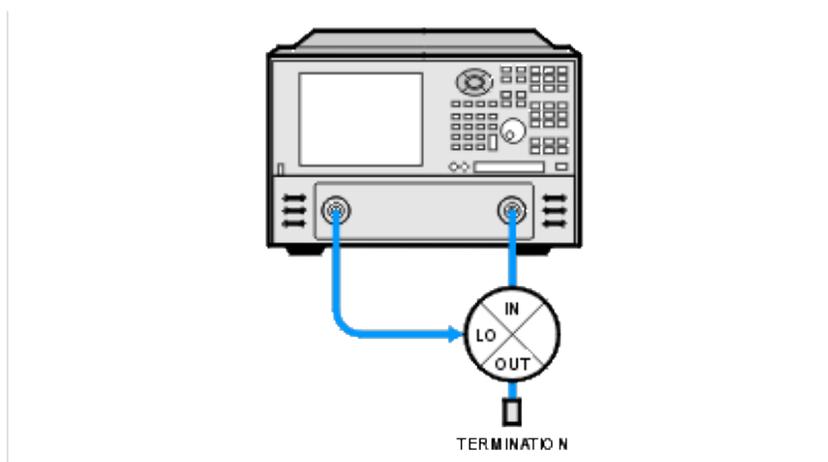
How to Measure Isolation

Use the following setups to measure the isolation of a mixer:

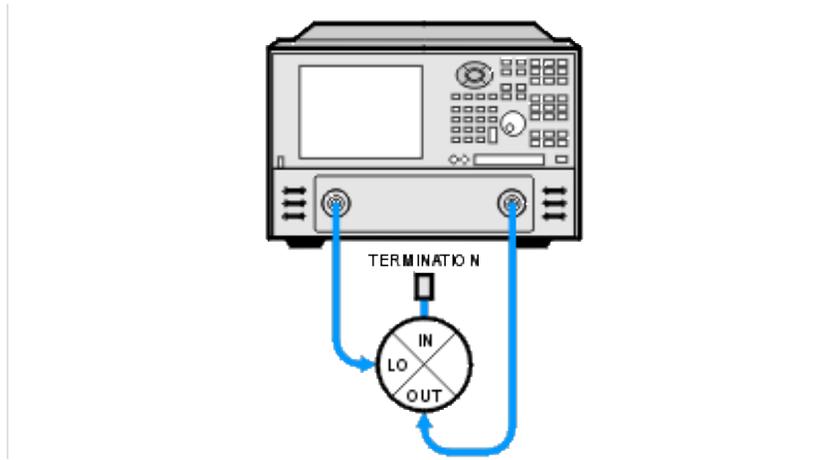
Note the following:

- The Input to Output isolation is very dependent on the LO power level. Isolation should be measured with the LO power at its normal operating level.
- Each of the ports not being tested should be terminated with an impedance typical of actual operation. This may not always be the characteristic impedance, Z_0 (usually 50 or 75 ohms). For example, if the OUT port of a mixer is intended to be directly connected to a filter, then this filter should be used when measuring the LO-to-IN feedthrough.

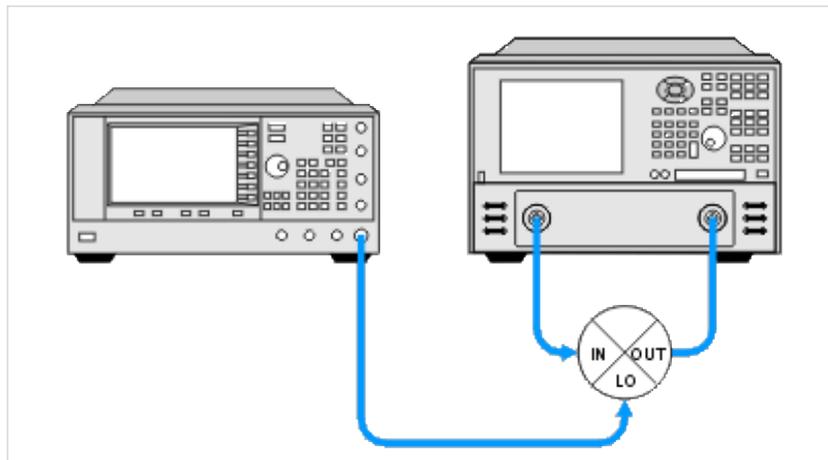
LO-TO-IN ISOLATION



LO-TO-OUT ISOLATION



IN-TO-OUT ISOLATION



Measuring Converters vs. Mixers

Measuring IN-to-OUT feedthrough of a converter is identical to that of a mixer. The IN-to-OUT feedthrough is generally very small for a converter due to the inclusion of an IF filter in the device. Because of this, the measurement may require the PNA to have increased [dynamic range](#).

Measuring LO leakage (LO-to-OUT and LO-to-IN) of a converter requires a different technique because the LO port is typically not accessible:

- The PNA can be tuned to the frequency of the LO signal and either the OUT or IN port connected to the PNA receiver port. The PNA source port is not connected.
- A spectrum analyzer can be connected to either the OUT or IN port and tuned to the frequency of the LO

signal.

Harmonic Distortion

- [What is Harmonic Distortion?](#)
- [Why Measure Harmonic Distortion?](#)
- [How to Measure Harmonic Distortion](#)
- [Measurement and Accuracy Considerations](#)

See other Frequency Converting Device Measurements

What is Harmonic Distortion?

Harmonics are multiples of any signal appearing at the mixer input and also multiples of the LO input. The distortion of the mixer's output characteristics caused by these harmonics is referred to as harmonic distortion. Harmonic distortion is caused by non-linearities in the device.

Harmonics are NOT signals created by two or more signals interacting (mixing); these signals are known as intermodulation products, which result in intermodulation distortion.

Why Measure Harmonic Distortion?

- It can degrade the performance of devices connected to the output of the mixer.
- The harmonics can also mix with other signals present in the mixer, adding to the intermodulation distortion of the mixer.

How to measure Harmonic Distortion

The harmonics can be measured using the PNA with [Frequency Offset](#) (option 80). The frequency of the LO to the mixer is set to zero and multiplier of the RF input is used to set the IF frequency (the harmonic). The equipment setup is shown below.

Since harmonics are specified in dBc, the fundamental RF and both the second and third harmonics are measured and the differences calculated. Multiple channels can be used to do this.

1. Connect the equipment.
2. Setup the measurement for calibration. See also [Measurement and Accuracy Considerations](#).
Use three channels and [frequency offset mode](#):

Channel 1 = F1 to F2

Channel 2 = F1 to 2F2 (frequency offset mode, multiplier = 1)

Channel 3 = F1 to 3F2 (frequency offset mode, multiplier = 1)

- Perform a source power calibration and receiver power calibration over the entire frequency range. See [Measurement and Accuracy Considerations](#).
- Reduce the frequency span and increase the frequency offset multiplier on Channels 2 and 3:
Channel 2 = F1 to F2 (frequency offset mode, multiplier = 2)

Channel 3 = F1 to F2 (frequency offset mode, multiplier = 3)

Note: Because the frequency span has been changed from that used for calibration, the source and receiver calibrations will be interpolated.

- Connect the DUT, make the measurement, and calculate the harmonic response:
Set up markers on Channels 1, 2 and 3, and determine the difference between the marker values to get the dBc value of each harmonic.

Channel 1 - Channel 2 = 2nd harmonic (dBc)

Channel 1 - Channel 3 = 3rd harmonic (dBc)

Note: Be sure to set the markers to the appropriate stimulus. Channel 2 markers should be set to twice the frequency of Channel 1 markers. Channel 3 markers should be set to three times the frequency of Channel 1 markers.

Measurement and Accuracy Considerations

Equipment Setup Considerations

- A filter must be used at the input of the mixer to remove the PNA source harmonics.

Return Loss and VSWR

- [What are Return Loss and VSWR?](#)
- [Why Measure Return Loss and VSWR?](#)
- [How to Measure Return Loss and VSWR](#)

[See other Frequency Converting Device Measurements](#)

What is Return Loss and VSWR?

Return loss and VSWR are both linear reflection measurements, even when testing frequency conversion devices, because the reflected frequency is not converted. These measurements are essentially the same as for filters and amplifiers. Learn more about [Reflection Measurements](#).

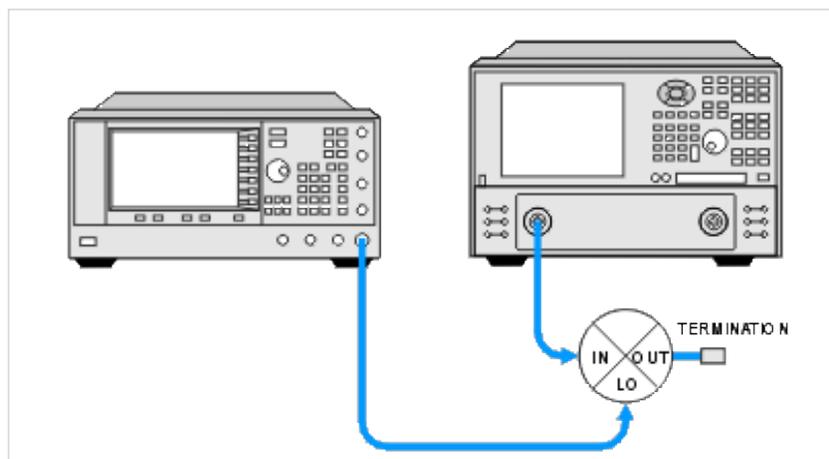
Why Measure Return Loss and VSWR?

Devices which have poor return loss and VSWR result in loss of signal power or degradation of signal information.

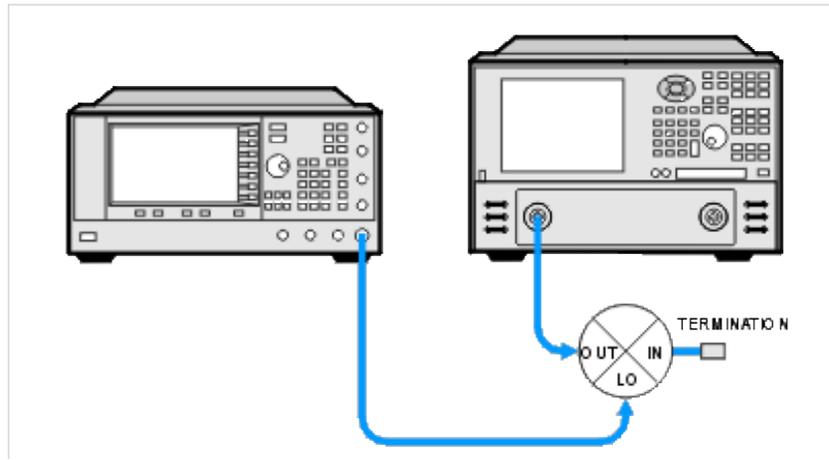
How to Measure Return Loss and VSWR

Setup the PNA measure return loss and VSWR as you would any two-port device. Connect your frequency converting device as shown in the following diagrams:

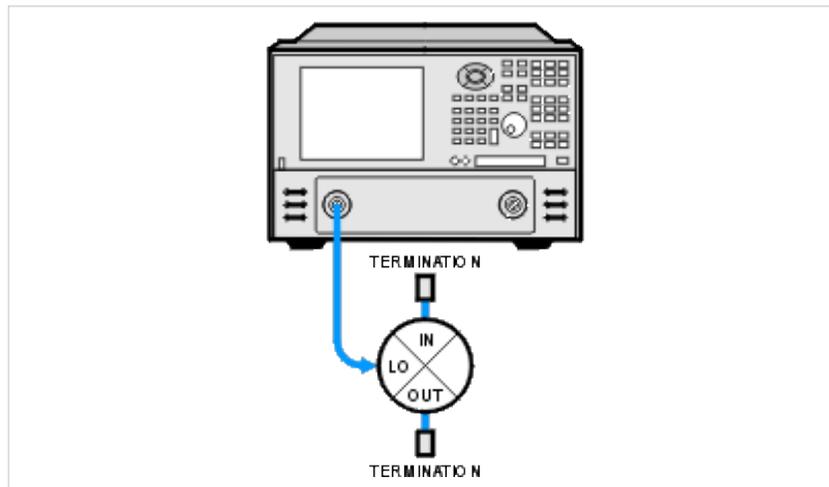
RETURN LOSS AND VSWR OF MIXER INPUT PORT



RETURN LOSS AND VSWR OF MIXER OUTPUT PORT



RETURN LOSS AND VSWR OF MIXER LO PORT



Gain Compression for Amplifiers GCA (Opt 086)

- [What's New](#)
- [Features, Requirements, and Limitations](#)
- [Gain Compression Concepts](#)
- [Understanding the GCA Displayed Traces](#)
- [Gain Compression Parameters](#)
- [Compression Methods](#)
- [Acquisition Modes](#)
- [Using Gain Compression App](#)
 - [Frequency tab](#)
 - [Power tab](#)
 - [Compression tab](#)
 - [Safe Sweep Mode dialog](#)
- [Compression Analysis](#)
- [Saving GCA Data](#)
- [GCA Measurement Tips](#)
- [Macros](#)

See Also

- [Gain Compression for Converters](#)
- [GCA Calibration](#)
- [Programming commands](#)
- **App Note** [Amplifier Linear and Gain Compression Measurements](#)

Other PNA Applications

Features, Requirements, and Limitations

Features

- Fast, easy, and complete Gain Compression measurements for amplifiers.
- Many [compression parameters](#) to choose from, including gain, input power at compression, output power at compression, input match, and compression level.
- Several [compression methods](#) to choose from, including deviation from linear gain, deviation from max gain, back-off, and X/Y, and compression from saturation.
- Three [acquisition methods](#) to choose from: Power per Freq, Freq per Power, and SMART Sweep
- [SMARTCal Calibration Wizard](#) to guide you through Full 2-Port or Enhanced Response calibration, plus Source Power calibration.
- [Compression Analysis](#) allows traditional power sweep view at a selected frequency.
- [Receiver Leveling](#) provides continuous source power accuracy.
- Supports Wideband (NOT Narrowband) Pulse measurements using the new integrated [Pulse setup dialogs](#).

Requirements

- Option 086 (software option only) [must be enabled](#).
- When performing an optional calibration:
 - ECal module or Calibration Kit
 - Power meter/sensor

Limitations with GCA

- Number of points limited to 20,001 for two-dimensional acquisitions, 10,000 points for SMART Sweep.
- Standard CW power sweep is NOT supported in a Gain Compression channel.
- Independent IFBW, Power Levels, or Sweep Time in a [segment table](#) is NOT supported.
- Stepped sweep mode only.
- Linear, Log, and Segment frequency sweep modes only.

The following PNA Features are **NOT** Available in a Gain Compression channel:

- [Unratioed receiver measurements](#) (A, B, R)
- [ECal User Characterization](#)
- Some [Fixturing Features](#)

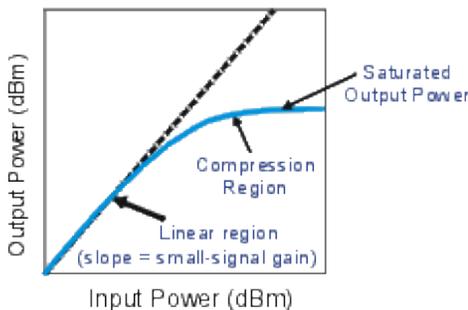
- [FOM](#) or [FCA](#)
- [External Test Set Control](#) (Option 551)
- [Interface Control](#)
- [IF Path Configuration](#)
- [Time Domain](#)
- [Balanced measurements](#)
- Save [Auto Formatted Citifile](#) data.
- Narrowband Pulse measurements using the [Integrated Pulse App](#)

Gain Compression Application Concepts

What is Gain Compression

An amplifier has a region of linear gain, where the gain is independent of the input power level. This gain is commonly referred to as small signal gain. As the input power is increased to a level that causes the amplifier to approach saturation, the gain will decrease. The 1 dB gain compression is defined as the input power level that causes amplifier gain to drop 1 dB relative to the linear gain.

You can quickly measure the gain compression using a [compression marker](#) on a power sweep trace.



Terms used in GCA

Linear Power Level The specified input power that yields linear gain (also known as 'small-signal gain') in the amplifier.

Reference gain The measured gain that is used as a reference for determining compression level. The [Compression Method](#) that is used could cause this value to be different.

Compression level The specified amount of gain reduction from the reference gain.

Target gain The gain at the specified compression level. Although this term does not appear in GCA,

it is important to understand when discussing the various compression parameters.

For example, when using [Compression from Linear Gain](#) method with the following settings:

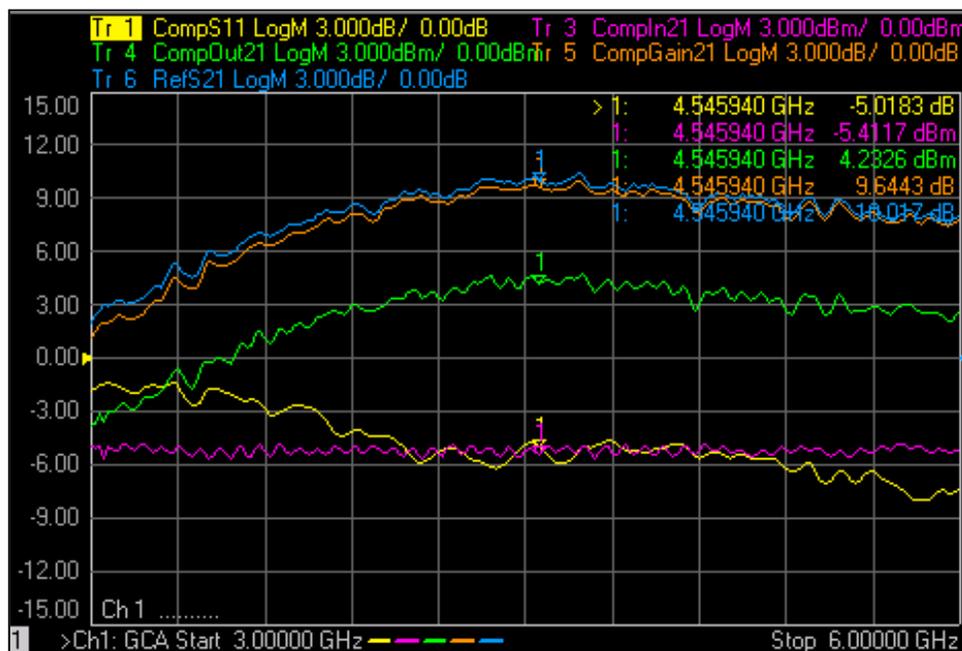
- Linear gain (measured at Linear Input power) = 10.2 dB
- Compression level (specified) = 1 dB
- Target gain = 9.2 dB

This is called 'Target' gain because GCA will search for the closest measured gain to 9.2000 dB. It may not measure this gain exactly.

Compression point The operating point at which the measured gain is closest to the Target Gain. All [compression parameters](#) report data for this operating point.

Understanding the GCA Displayed Traces

One of the most important concepts to remember with GCA is that, each frequency data point represents many measurements using different input power levels.



Some things to notice about how GCA displays **compression** data:

1. The X-axis values are ALWAYS frequency. Imagine behind each frequency data point, a traditional power sweep curve with corresponding measurements and calculations to find the specified compression point.
2. The Y-axis values are always reported at the [compression point](#). The value that is displayed depends on the

compression parameter that you choose. The [S-parameters](#) that are displayed in a GCA channel are always measured at the [linear and reverse](#) power level.

Example: Five of the six GCA [compression parameters](#) are displayed in the above image. The missing trace, [DeltaGain21](#) is discussed below.

- Markers are placed at 4.549 GHz for all of the parameters.
- **Tr 3 Compln21** (Input power at the compression point) shows the marker value to be **-5.4117 dBm**. This is the power into the DUT that was required to achieve the compression point. Notice that this is about the same input power required to achieve the specified compression at ALL frequencies.
- **Tr 5 CompGain21** (Gain at the compression point) shows the marker value **9.6443 dB**. This is the measured gain at the compression point.
- To see the gain at a different input power at this frequency, use the [Compression Analysis](#) feature.

Gain Compression Parameters

There are several Gain Compression parameters, as well as standard S-parameters and ADC parameters, that can be measured in a GCA channel.

How to add GCA Parameters

First, create a GCA channel. [Learn how](#). The default parameter is S21.

Using front-panel HARDKEY [softkey] buttons

1. Press **TRACES**
2. then **[New Trace]**
3. then select a parameter

Using a mouse with PNA Menus

1. Click **Trace/Chan**
2. then **New Trace**
3. then select a parameter

Programming Commands

Linear S-Parameters

For convenience, the standard S-parameters are offered in a GCA channel. S11 and S21 are measured at the specified Linear Input level. S22 and S12 are measured at the specified Reverse power level.

Parameter	Description	When Measured
S11	Input Match	Always
S21	Gain	Always
S22	Output Match	See Reverse
S12	Reverse Isolation	See Reverse

ADC Parameters

Four ADC analog-to-digital converter measurements are offered in a GCA channel:

- **AI1** and **AI2** are measured at the specified Linear Input level.
- **CompAI1** and **CompAI2** are measured at the specified compression point.

These DC measurements, along with [Equation Editor](#), allow you to make PAE measurements at the Linear Input level and compression point.

These measurements are made at pins 7 and 8 of the PNA [Power IO connector](#).

[Learn more about ADC measurements.](#)

Compression Parameters

Note: The following table assumes: DUT **Input** = PNA **port 1** and DUT **Output** = PNA **port 2**.

When the Port mapping is different, the parameters in GCA are updated accordingly. For example, with Input = port 2 and Output = port 1, then "CompIn12" would be displayed.

The raw data for these parameters are always measured.

Parameter	Description
CompIn21	Input power at the compression point.
CompOut21	Output power at the compression point.
CompGain21	Gain at the compression point.
CompS11	Input Match at the compression point.
RefS21	Linear Gain value used to calculate the compression level. This is calculated differently depending on the compression method.
DeltaGain21	CompGain21 MINUS Linear Gain (in Log Mag format). This trace can be used to learn a lot about the DUT compression point. Learn more.

Compression Methods

GCA offers the following methods to find the compression point of an amplifier using GCA:

Compression from Linear Gain

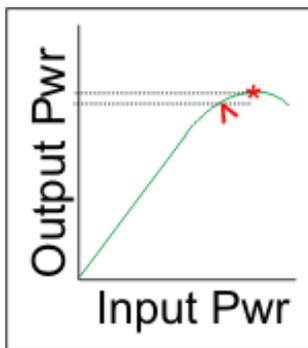
The Reference Gain is measured using the specified Linear (Input) Power Level. The Target Gain is calculated as the Linear Gain minus the specified Compression Level. For example 8.3 dB - 1 dB = 7.3 dB.

Compression from Max Gain

The linear region of an amplifier gain may not be perfectly linear. The highest gain value that is found at each frequency is used as the Reference (S21) Gain. The [Target Gain](#) is found in the same way as Compression from Linear Gain.

Compression from Saturation

This method is used to better find the compression point when measuring amplifiers with non-linear gain as shown in the following image:



The Max power out value * is found at each frequency. Then input power is lowered until the output power decreases by the specified 'From Max Pout' value. This is the compression point. ^

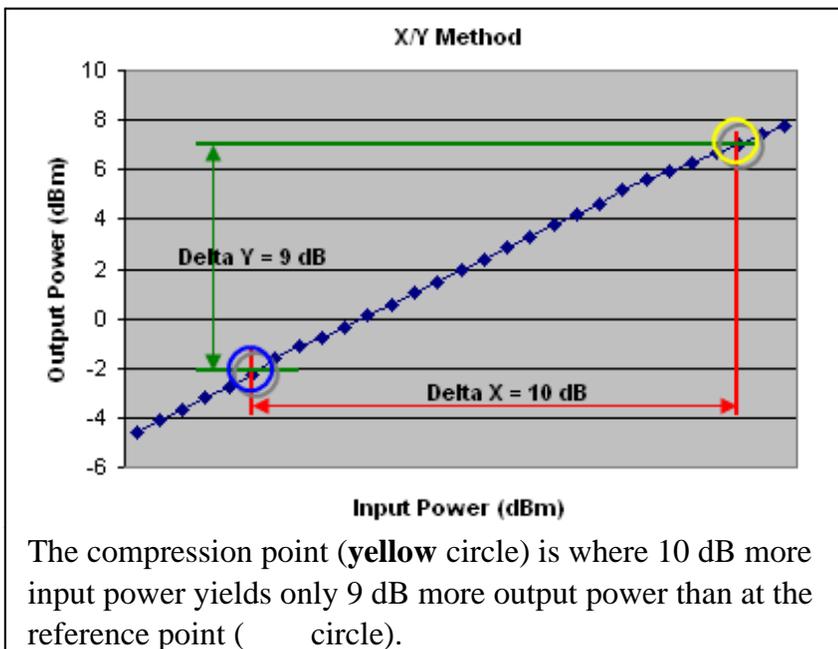
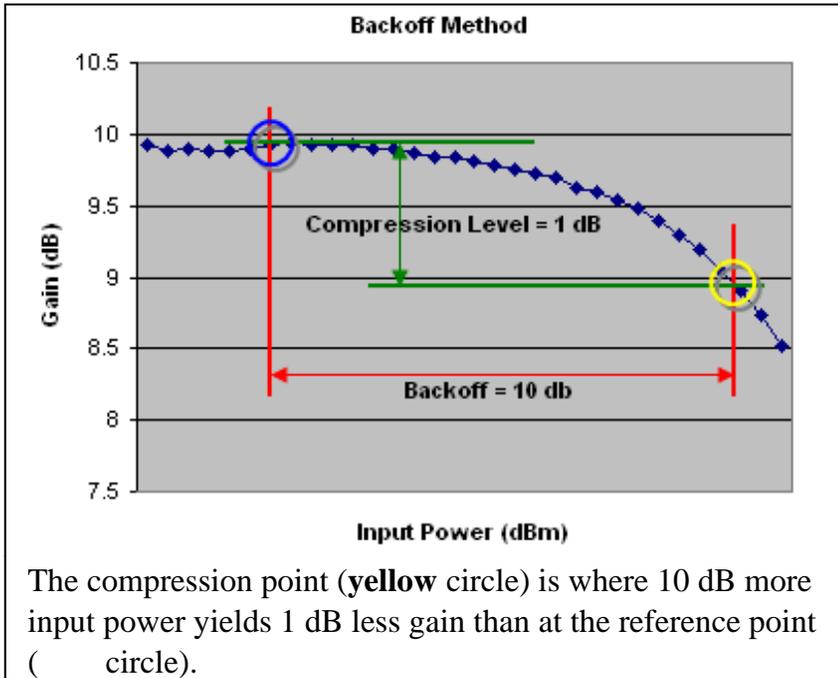
Backoff and X/Y method

These two compression methods are very similar.

- Both methods specify a difference in input power (X axis) between the linear region and compression point.
- For the Y-axis difference:
 - **Backoff method** specifies Compression Level which is a difference in **Gain**.
 - **X/Y method** specifies Delta Y which is a difference in **Output Power**.

GCA searches for these points differently for [2D sweeps](#) and [SMART sweep](#).

The following images show how Backoff and X/Y method is calculated at ONE frequency.



Acquisition Modes

The GCA offers three modes for data acquisition: Two 2D sweep modes, and [SMART sweep](#).

To see a traditional power sweep at a single frequency, use the Compression Analysis feature. [Learn more.](#)

2D (two-dimensional) Sweeps

This is the easiest method to understand, and the least efficient for finding the compression point. Both 2D sweep modes work as follows:

1. All GCA measurements begin by measuring S-parameters at the specified Linear Power level. Reverse parameters are measured ONLY if Full 2-port calibration is applied or if a reverse parameter is displayed. [Learn more about Cal choices.](#)
2. Gain measurements are then made at ALL of the specified frequency and power values. Although these are conceptually 2-Dimensional sweeps, a single sweep is constructed in firmware. [See Data Points Limit.](#)
3. After data has been measured, a search is performed to find the compression point. You can choose to interpolate between the two measured points closest to the target gain. [Learn more.](#)

As each sweep is performed, dots are plotted next to the **Ch** indicator in the lower left corner of the display to indicate progress for the current sweep.

Note: For [Backoff and X/Y compression method](#), GCA does not verify that the specified Start - Stop power range is at least the size of the specified Backoff or X value. The closest compression point is always reported.

Note: [SMU Hardware List](#) trigger mode is NOT supported in GCA 2D sweeps.

2D Sweep Modes

- **2D Sweep Power per Frequency** - Input power is stepped from [Start to Stop](#) at each specified frequency. From the following example you can see that the device is exposed to the highest power level (p3) at the first frequency (f1). This could heat the device early in the measurement and affect compression results.

The following examples show (**frequency, power**) values for three frequency points and three power points, resulting in a total of 9 measurements:

1	2	3	4	5	6	7	8	9
f1,p1	f1,p2	f1,p3	f2,p1	f2,p2	f2,p3	f3,p1	f3,p2	f3,p3

- **2D Sweep Frequency per Power** - Frequency is swept from start to stop at each specified power level as follows:

1	2	3	4	5	6	7	8	9
f1,p1	f2,p1	f3,p1	f1,p2	f2,p2	f3,p2	f1,p3	f2,p3	f3,p3

Viewing and Saving 2D Data

It is NOT possible to plot ALL of the 2D measurement data on the PNA display. However, it can be saved to a *.csv file and then read into an Excel spreadsheet. The initial S-parameter measurement data is not saved to this file. [Learn more.](#)

You can also view on the PNA all power sweep information at a selected frequency using the [Compression Analysis](#) feature.

SMART Sweep

SMART Sweep is usually the fastest and most accurate method to measure Gain Compression. Unlike the 2D acquisition modes which measure all of the specified frequency / power points, SMART Sweep performs a series of power search iterations. At each frequency, an 'intelligent guess' of input power is made to find the compression level that is within tolerance. This guess is further refined with each successive power search iteration sweep.

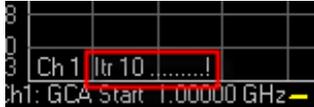
SMART Sweep continues to iterate until one of the following conditions occur:

1. ALL data points are within tolerance. When the compression level for a data point achieves the specified tolerance, it continues to be measured and input power changed to improve the measurement within tolerance.
2. The specified compression level can NOT be achieved for the remaining frequencies that are not in tolerance. Either the Start power is too high or the Stop power is too low.
3. Maximum iterations have been achieved. If a measured gain is not within the specified tolerance before the specified **Max** number of **Iterations** has been reached, then the **last** power reading is used as the compression point.

The Iteration Counter, Dots, and Bangs(!)

Next to the Ch indicator, in the lower left corner of a GCA window, the following annotation appears:

- An **iteration counter** is incremented each time input power is adjusted.
- A **dot** appears when another 10% of the frequency points are within tolerance.
- **!** (bangs) are displayed after the last iteration. Each bang represents 10% of the data points that are NOT within tolerance.



SMART Sweep and Compression Method

The intelligent guess process works differently depending on the compression method. This is important because Backoff and X/Y compression methods subject the DUT to significant changes in input power during an iteration sweep. This can affect the DUT and the measurement results.

[Learn all about Backoff and X/Y compression methods.](#)

ALL GCA measurements begin by measuring S-parameters at the specified Linear Power level. Reverse parameters are measured ONLY if Full 2-port calibration is applied or if a reverse parameter is displayed. [Learn more about Cal choices.](#)

- **Backoff and XY** Because both compression methods specify the separation between the "linear" region and the "compressed" region, each iteration requires a single sweep at **two dramatically** different power levels over the same frequency range. The first half of the sweep measures the DUT at the **Backoff** or **X** power level. The second half of the sweep measures the DUT at the compressed power level, specified by the [Start and Stop](#) power range. At the beginning of the second half, the power level rises by the **Backoff** or **X** value. The specified [Settling Time](#) is applied at this point to allow the DUT time to react to this significant change in power level. [Safe Sweep](#) does **NOT** minimize this change in input power. However, Safe Sweep with Backoff and XY methods **DOES** prevent the DUT from being exposed to too much input power.
- **Compression From Linear Gain** After the reference gain is measured at the linear input power, the next iteration measures the DUT at a higher power level which attempts to push the DUT well into compression. Subsequent sweeps, depending upon the compression level of the DUT, either increases or decreases the power in order to reach the desired compression level. Usually, by the third iteration sweep, a curve-fit algorithm is utilized to precisely find the compression point.

Note: The DUT can be subject to significant changes in power from one iteration sweep to the next. This can be minimized by the use of [SAFE Sweep](#) and careful selection of the corresponding settings.

- **Compression from Max Gain** The maximum gain that is found at each frequency is stored and used to calculate the compression point. SMART Sweep does NOT perform extra iterations to search for the maximum possible gain of the amplifier at each frequency.
- **Compression from Saturation** The maximum power out that is found at each frequency is stored and used to calculate the compression point. SMART Sweep does NOT perform extra iterations to search for the maximum possible power out of the amplifier at each frequency.

Using the Gain Compression Application

The following is a general procedure for performing a GCA measurement. The challenge with GCA is configuring a measurement that yields the true compression performance of YOUR DUT. This requires knowledge of the Gain Compression settings and knowledge of the DUT.

See specific dialog boxes below.

1. Disconnect the DUT if preset or default power levels may damage the PNA or DUT.
2. [Preset](#) the PNA, or configure a suitable [User Preset](#) that will be safe in case the DUT is connected.
3. Create a GCA channel. [Learn how](#). The default trace is S21.
4. Start [GCA Setup dialog](#) and configure the measurement settings based on the DUT, adapters, attenuators, booster amplifiers, and fixtures to be used in the measurement.
5. Save the [instrument state](#) (optional).
6. Connect DUT and apply bias and RF power as appropriate. The default measurement for a GCA channel is S21 (amplifier gain). Inspect the gain measurement to ensure the DUT is operating as expected.
7. Add GCA compression parameter traces. [Learn how](#).
8. Adjust the measurement settings to yield satisfactory compression parameters. [See GCA Measurement Tips](#).
9. Start and complete the [GCA Calibration wizard](#).

How to start the Gain Compression Setup dialog

To provide quicker access, use the Setup softkey. [Learn how](#).

Using front-panel HARDKEY [softkey] buttons

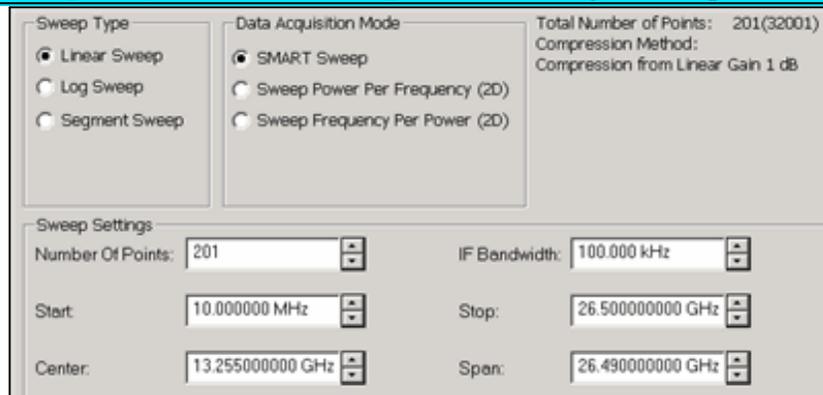
1. Press **FREQ**
2. then **[Gain Compression Setup]**

Using a mouse with PNA Menus

1. Click **Stimulus**
2. then **Frequency**
3. then **Gain Compression Setup**

Programming Commands

Frequency tab - Gain Compression -dialog box help



Configures the frequency settings over which Gain compression is to be measured, as well as the measurement method.

Sweep Type

Choose a method in which to sweep frequency: Linear, Log, and Segment Sweeps. This setting applies to all data acquisition modes.

Notes

- Log and Segment Sweep are NOT available on [GCX](#).
- CW Sweep is NOT available in GCA. However, to see a traditional power sweep at a single frequency, use the [Compression Analysis](#) feature.
- To use CW Sweep in GCX, set all ranges to **Fixed** on the [Mixer Frequency tab](#). This has the same effect as setting all ranges to Start = Stop Frequency.

Segment Sweep Notes (GCA ONLY)

- The segment table shown on the dialog is '**READ-ONLY**'.
- Learn how to [Create and edit the Segment Sweep table](#).
- **Independent IFBW** and **Power** are NOT available.
- [X-axis point spacing](#) is available beginning with A.09.10.

Data Acquisition Mode

Specifies HOW the gain compression data is collected.

SMART Sweep

- At each frequency, input power is 'intelligently' adjusted to find a measured gain equal to the target gain.
- Faster and more accurate than 2D sweeps to measure Gain Compression point at a number of frequencies.
- [Learn ALL about SMART Sweep](#)

2D (two-dimensional) Sweeps

- **Sweep Power per Frequency** Performs a series of power sweeps at each successive frequency.
- **Sweep Frequency per Power** Performs a series of frequency sweeps at each successive power level.
- [Learn ALL about 2D sweeps](#)

Sweep Settings

Click each to learn more about these settings.

- [Number of points](#) Number of frequency points to measure. The Frequency points may be limited due to the number of specified Power points. [See Data Points Limit](#).
- [IF Bandwidth](#) Set this value to yield acceptable trace noise when measuring gain at the linear power level. This level of noise contributes directly to the accuracy of compression point. A lower value (narrower IFBW) allows for more accurate, but slower, measurements. [See GCA Measurement Tips](#) to see how to best set IFBW.
- [Start / Stop, Center / Span](#) frequencies. Set the frequency range over which to measure Gain compression.

Data Points Limit

The maximum number of measurement data points depends on Acquisition method and Compression method as follows:

	SMART sweep	2D sweep
Compression method	Number of frequency points is reduced to ensure the total number of data points does not exceed the specified limit.	Number of power points is reduced to ensure the total number of data points does not exceed the specified limit.
<ul style="list-style-type: none"> • Compression from linear gain 	Data points = freq points Max = 20,001	Data points = (freq. points) * (power points) Max power points = 2,001 Max data points = 20,000
<ul style="list-style-type: none"> • Compression from max gain 	Data points = freq points Max = 20,001	
<ul style="list-style-type: none"> • X/Y and Backoff 	Data points = 2 * freq points Max = 20,001	

Note: Although the dialog box will allow you to enter any number of frequency or power points, the values are checked when **OK** or **Apply** is pressed. If a limit is exceeded, the relevant data points are reduced to the maximum allowable number **without warning**.

Power tab - Gain Compression dialog box help

Power On (All Channels) Total Number of Points: 201 (32001)

DUT Input Port: Input Port: Port 1 DUT Output Port: Output Port: Port 2

Linear Input Power: -25.00 dBm Reverse Power: -5.00 dBm

Source Attenuator: 0 dB Source Attenuator: Auto 0 dB

Receiver Attenuator(A): 0 dB Receiver Attenuator(B): 0 dB

Source Leveling Mode: Internal Source Leveling Mode: Internal

Power Sweep:

Start (Min) Power: -25.00 dBm

Stop (Max) Power: -5.00 dBm

Power Points: 21

Power Step: 1.000 dB

Path Configuration...

Configures RF power and Power Sweep settings for Gain Compression measurement.

Power ON (All channels) Check to turn RF Power ON or clear to turn power OFF for all channels.

Input Port

Select the PNA port that is connected to the DUT Input.

Linear Power Level The input power that yields the linear gain of the DUT. The linear gain is used as the reference gain when calculating the **Compression from Linear Gain**. Input match is also measured at this power level.

Source Attenuator Specifies the attenuator setting associated with the port connected to the input of the DUT. This attenuator will affect the range of available power into the DUT [Learn more about Source Attenuation](#).

All PNA channels in continuous sweep must have the same attenuation value. [Learn more](#).

Receiver Attenuator Specifies the attenuator setting for the receiver associated with the output of the DUT. When the power into the receiver test port is around +10 dBm, the PNA receiver may be in compression. However, with receiver attenuation, lower input power levels may become too noisy to make accurate power measurements. In this case, lower IFBW to reduce noise. [Learn more about Receiver Attenuation](#).

Source Leveling Specifies the leveling mode. Choose Internal or Receiver R1. [Learn how to configure Receiver Leveling](#). Open Loop should only be used when doing [Wideband Pulse measurements](#).

Output Port

Select the PNA port that is connected to the DUT Output.

Reverse Output Power Sets power level into the output of the DUT for reverse sweeps. Port power is automatically uncoupled.

Reverse power is applied to the DUT ONLY under the following conditions. Otherwise, this setting is ignored.

- When Linear Output Match or Linear Reverse Isolation parameters are requested.
- When Full 2-port correction is used. You can perform a full 2-port cal and downgrade to an Enhanced Response Cal to prevent reverse power from being applied to the DUT. [Learn more](#).

Source Attenuator Specifies the attenuator setting for the port connected to the DUT output. This setting will affect the range of available power at the DUT output port.

Receiver Attenuator Specifies the attenuator setting for the receiver associated with the DUT output port.

Source Leveling Specifies the leveling mode. Choose from: Internal (normal operation) or Open Loop (used only for [Wideband Pulse measurements](#)).

Power Sweep

Power Points Number of power points to measure for 2D acquisition modes. The Power Points may be limited due to the number of frequency data points. [See Data Points Limit](#). This setting is NOT available in SMART Sweep, which uses only enough power points to find the specified compression level.

Start and Stop Power

- **2D sweep** In Backoff, X/Y, and Compression from Max Gain methods, sets the range of power levels that are applied to the DUT to find BOTH the [Reference Gain](#) and [Compression point](#). Make sure this range is wide enough to include both. For example, if the Backoff level is 10 dB, then the power range must be greater than 10dB. Otherwise, GCA will report a compression value using the closest reference gain and compression point, which may be inaccurate. In Compression from Linear Gain, the reference gain is measured at the Linear Power Level, so the Start and Stop power levels are used to find the compression point.
- **SMART sweep** Sets the range of power over which GCA will search for the compression point. The reference gain is found using the Linear Power Level, Backoff, and X values, depending on the Compression Method. To reduce the number of iterations that are required to find the compression point, limit the Start / Stop power range to the input levels that will achieve compression. Do not include the linear region.

Note: If your DUT requires more input power to achieve compression below 3.2 GHz, use the PNA-X **Hi-power mode**, available from the [RF Path Configuration](#) dialog. The disadvantage to this is higher harmonic content.

Power Step (Size) Calculated value from current Start, Stop, and Points settings. This setting can NOT be changed directly.

Path Configuration click to launch the [RF Path Configuration](#) dialog.

Compression tab - Gain Compression dialog box help

Compression Method

[Learn ALL about these Compression Methods](#)

- **Compression from Linear Gain** The specified compression level is measured from the linear gain. The linear gain is measured using the **Linear Power Level** that is specified on the [Power tab](#).
- **Compression from Max Gain** The specified compression level is measured from the maximum gain level. In SMART sweep, the Max Gain value is updated as each iteration occurs. To increase the chances of measuring the actual maximum gain of the amplifier, [Safe Sweep](#) should be invoked using low Coarse and Fine increments.
- **Compression from Back Off** This compression method uses the Compression Level and Back Off values for finding the compression point.
- **X/Y Compression** This compression method uses the specified parameters (X and Y) as the criterion for finding the compression point.
- **Compression from Saturation** Similar to Compression from Max Gain, except the specified compression level is measured from the maximum power out level. Use this method to better find the compression point when measuring amplifiers with non-monotonic gain. In SMART sweep, the Max power out value is updated as each iteration occurs. To increase the chances of measuring the actual maximum power out of the amplifier, [Safe Sweep](#) should be invoked using low Coarse and Fine increments

SMART Sweep

[Learn ALL about Smart Sweep.](#)

Tolerance Specifies an acceptable range for measuring the compression level. Reducing this value can significantly increase the number of iterations that are required to find the compression point.

Maximum Iterations Specifies the maximum number of power search iterations SMART Sweep is allowed. Reducing this value can cause SMART sweep to terminate before all compression levels are found to within the specified tolerance.

Show Iterations When checked, the compression parameter traces are updated at the completion of each power search iteration. When cleared, compression parameter traces are updated when SMART Sweep completes the power search iteration process.

2D Sweep - Compression Point Interpolation

When a 2D Sweep is selected (on the [Frequency tab](#)), check this box to calculate and display interpolated compression traces.

The [Target gain](#) is calculated using a complex linear ratio between the two closest measured values. All compression parameters are then interpolated using this same ratio.

Clear the box to display compression parameters for the closest compression point, either high or low, to the level specified in the Compression Method setting.

End of Sweep Condition Specifies the power level applied to the DUT at the completion of a GCA measurement.

GCA performs numerous power and frequency sweeps on the DUT during the overall measurement process. This setting has no affect on these intermediate sweeps. This setting only applies at the end of the very last sweep in the GCA channel.

In addition, this setting applies ONLY to the GCA channel. All other channels operate independently of this setting. Therefore, the power applied to the DUT after all channels have been measured may be different from this setting.

Choose from:

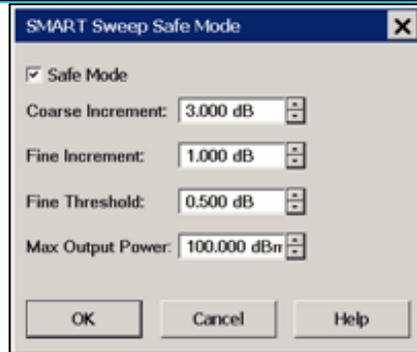
- **Default** Use the default PNA method. [Learn more.](#)
- **RF OFF** RF power is turned off when GCA completes a measurement cycle.
- **Start Power** RF power is set to the start power level.
- **Stop Power** RF power stays at the stop power level.

Settling Time

Used ONLY in SMART Sweep when Back Off or X/Y compression algorithms are selected.

This setting allows additional dwell time when the input power changes from the back-off level to the compression level. [Learn more.](#)

Safe Sweep Mode dialog box help



For use with SMART Sweep ONLY.

When enabled, Safe Sweep increases the input power to the DUT by the specified amounts, allowing the compression point to be achieved gradually. While this will increase the number of iterations required to achieve compression, it also minimizes the possibility of driving the DUT too far into compression.

Note: Safe Sweep does **NOT** minimize the dramatic change in input power with Backoff and XY method. However, Safe Sweep with Backoff and XY methods **DOES** prevent the DUT from being exposed to too much input power. [Learn more.](#)

Safe Mode (Enable) Check to enable Safe Sweep.

Coarse Increment Sets the maximum change in input power, up or down, which will be applied to the DUT from one iteration to the next. Default = 3.0 dB.

Without Safe Sweep, the maximum change in input power can be the entire Backoff or X value when using these compression methods.

Fine Increment Once the Fine Threshold has been achieved, this becomes the maximum change in input power, up or down, which will be applied to the DUT. Default = 1.00 dB

Fine Threshold Specifies the compression level in which Safe Sweep changes from the COARSE to the FINE increment. Default = 0.5 dB. This means that, by default, the PNA uses the Fine Increment adjustment when compression reaches 0.5 dB.

Max Output Power To protect the PNA from damage, when the PNA port that is connected to the DUT Output measures the specified value, the input power to the DUT is no longer incremented at that frequency. In these cases, the compression point would probably not be achieved.

Compression Analysis

Compression Analysis changes the current trace into a power sweep trace at a specified CW frequency .

The current parameter and acquisition method is unchanged. For example, with a CompGain21 trace displayed and SMART Sweep selected, enable Compression Analysis. The trace becomes a power sweep trace at the specified CW frequency. The Y-axis displays S21 Gain at each X-axis power point.

When Smart sweep is used, a complete power sweep is not performed, but only the data points that are required to find the compression point. To see a traditional power IN vs power OUT compression sweep, use one of the [2-D acquisition methods](#).

You can create PNOP or PSAT markers on a CompOut trace with Compression Analysis mode ON. [Learn more.](#)

How to perform Compression Analysis

With any [compression parameter](#) (such as CompGainS21) displayed:

Using front-panel HARDKEY [softkey] buttons

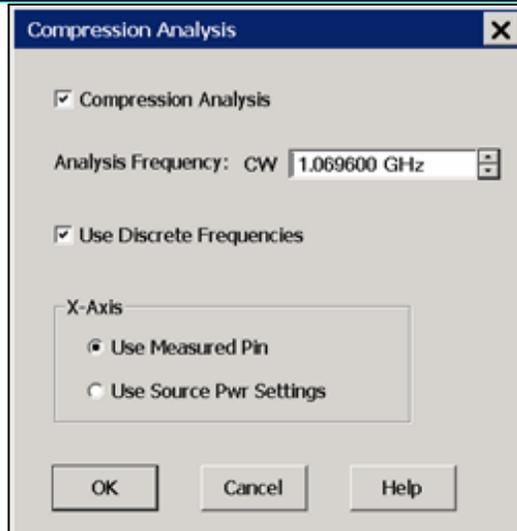
1. Press ANALYSIS
2. then [Compression Analysis]

Using a mouse with PNA Menus

1. Click **Marker/Analysis**
2. then **Compression Analysis**

Programming Commands

Compression Analysis dialog box help



Notes: When an S21 or S11 trace is active, any [compression parameter](#) (such as CompGainS21) must also be displayed.

Compression Analysis is NOT allowed for S12 or S22 traces.

Scroll up to [learn more about Compression Analysis](#).

Analysis Frequency: CW Enter a frequency to use for the compression analysis trace.

Compression Analysis Check to perform compression analysis. A compression trace is displayed at the Analysis (CW) Frequency.

Use Discrete Frequencies Check to allow Analysis Frequencies at only the discrete points where data is measured. Clear to allow Analysis CW Frequencies that are interpolated from the data points. Then select ANY CW frequency between the start and stop frequencies of the GCA channel.

X-Axis

- **Use Measured Pin** The X-axis displays the actual power that is applied to the DUT after match correction and R-channel drift correction.
- **Use Source Pwr Settings** The X-axis displays the power level of the stimulus.

Saving GCA Data

Beginning with PNA release A.08.20, GCA data can be saved to a *.csv file in both 2D and SMART Sweep modes (previously only 2D modes). Also, a Delta Gain, AI1, and AI2 columns have been added to the data. [Learn about ADC parameters](#).

How to save GCA data

With a GCA [Compression](#) trace active:

Using front-panel HARDKEY [softkey] buttons

1. Press **SAVE**
2. then [**Save Data As**]
3. **File Type = GCA Sweep *.csv**

Using a mouse with PNA Menus

1. Click **File**
2. then **Save Data As**
3. **File Type = GCA Sweep *.csv**

◀ **Programming Commands** ▶

Notes

- This data type can be read by spreadsheet programs, such as Microsoft Excel.
- Data from the last **complete** sweep is saved to the specified *.csv file.

- If calibration is turned **ON** when the file is saved, then all data is calibrated. Otherwise, raw data is saved.
- All *.csv data saves include a reference power level sweep at the beginning of each frequency data.

	A	B	C	D	E	F	G	H	I	J	K
1	Agilent Technologies_N5242A.US46290053.Z.08.04.12										
2	Date: Thursday, April, 10, 2008 14:15:28										
3											
4	Calibration State OFF*										
5	Num Freq	3									
6	Num Iterat	5									
7											
8	Frequency S11	Pin Data			Pout Data			S21		Delta Gain	
9	<Hz>	<LogMag (<Phase (D	<LogMag (<Phase (D						
10	4.00E+09	-4.6375	152.421	-25	0	-17.2892	-34.3579	7.71076	-34.3579	4.44E-16	1.52E-15
11	4.00E+09	-4.6154	152.473	-19.7657	-132.092	-12.0009	-166.651	7.76495	-34.5593	0.054092	-0.20141
12	4.00E+09	-4.61091	152.572	-16.7746	-54.5945	-8.99388	-88.9208	7.79073	-34.3263	0.069968	0.031649
13	4.00E+09	-4.6083	152.424	-13.7818	160.191	-6.00358	125.667	7.77822	-34.5231	0.067459	-0.16519
14	4.00E+09	-4.59279	152.327	-10.7716	152.283	-3.00766	117.78	7.76391	-34.5024	0.053151	-0.14452
15	4.25E+09	-5.43633	-125.845	-25	0	-15.7527	4.97897	9.24726	4.97897	0	4.20E-16
16	4.25E+09	-5.46401	-125.925	-19.979	-118.327	-10.727	-113.371	9.252	4.95641	0.004746	-0.02256
17	4.25E+09	-5.45105	-125.983	-16.978	-141.044	-7.71819	-136.062	9.25981	4.98213	0.012559	0.003156
18	4.25E+09	-5.47133	-126.249	-13.9661	80.5926	-4.7118	85.3736	9.24425	4.78099	-0.003	-0.19798
19	4.25E+09	-5.46864	-126.271	-10.9578	-68.4214	-1.74774	-63.6055	9.2101	4.81595	-0.03716	-0.16302
20	4.50E+09	-6.37248	-34.7728	-25	0	-14.521	52.0538	10.4789	52.0538	-8.88E-16	9.26E-16
21	4.50E+09	-6.377	-35.0287	-20.0535	-142.773	-9.59433	-90.7553	10.4691	52.0174	-0.00982	-0.03642
22	4.50E+09	-6.349	-34.9226	-17.0636	-165.799	-6.56493	-113.678	10.4987	52.1201	0.019756	0.066347
23	4.50E+09	-6.36875	-35.0064	-14.055	55.4364	-3.57836	107.509	10.4766	52.073	-0.00234	0.019178
24	4.50E+09	-6.36475	-34.91	-11.0519	-94.0431	-0.59745	-41.9732	10.4545	52.0699	-0.02445	0.016094

SMART Sweep data with 5 iterations and 3 frequency points. The yellow highlight is added here for readability.

When saving or recalling 2D data:

- When Linear Input Power EQUALS Start Power, then the number of data points (rows)/ freq = num power points.
- When Linear Input Power does NOT EQUAL Start Power, the number of data points (rows)/ freq = num power points + 1.
- Make these selections on the GCA/GCX [Power tab](#) dialog.

GCA Measurement Tips

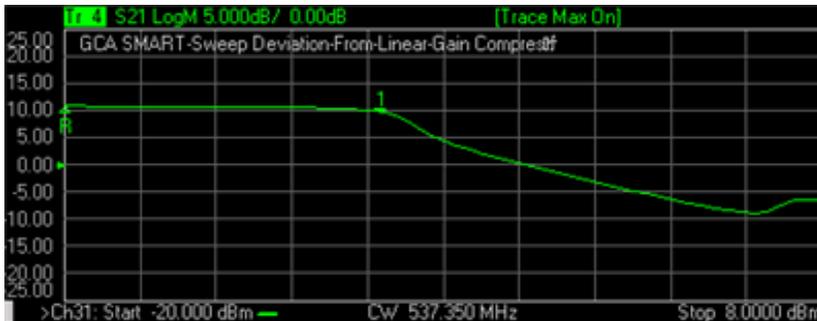
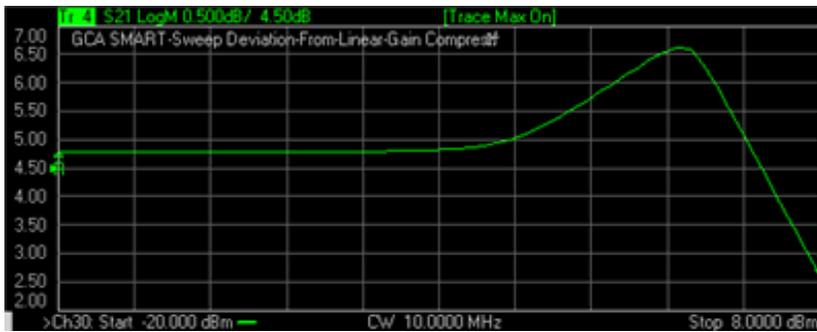
There are many settings in the Gain Compression Application. Here are a few tips when using GCA to learn as much as possible about the compression characteristics of your DUT in the most efficient manner.

DUT Compression Characteristics and GCA

Although GCA provides excellent results with a wide variety of amplifiers, it works best with amplifiers which have a monotonic compression curve. In some cases where the compression curve is not monotonic, for example if the amplifier gain expands before it compresses, the correct compression level may not be found.

To help a SMART sweep find the correct compression point, limit the Start and Stop power levels around the anticipated compression point. [Learn more.](#)

The following two power-sweep traces are examples of non-monotonic gain:



DeltaGain

A DeltaGain trace is the best way to see how closely GCA is actually measuring to the desired compression level. In addition, you can view the phase of DeltaGain to see the phase deviation between the [compressed gain](#) and the [reference gain](#). DeltaGain is calculated as:

- $\text{DeltaGain} = \text{Measured Gain (watts)} / \text{Ref Gain (watts)}$
- In LogMag format: $\text{DeltaGain} = (\text{Measured Gain}) - (\text{Ref Gain})$

With SMART Sweep, DeltaGain (in LogMag format) shows how soon certain frequencies achieve the specified tolerance. [Learn more.](#)

Some other settings which may be helpful:

- Trigger source: Manual allows you to analyze data and make adjustments while allowing the device to cool.
- Construct Limit Lines around the compression point at the tolerance level.

The following image shows a DeltaGain21 trace using SMART Sweep. The Limit Lines were added manually.



In the above image:

<p>Relevant Settings</p>	<p>Method = Compression From Linear Gain</p> <p>Compression level = 1</p> <p>Iteration Tolerance = 0.05 dB.</p> <p>Maximum Iterations = 10</p>
<p>Displayed Results</p>	<p>A data point on -1.00 indicates that, at that frequency, the exact compression level (1 dB) was measured.</p> <p>Several frequencies did not achieve the specified tolerance (0.05 dB) before the Max Iterations (10) was reached.</p> <ul style="list-style-type: none"> • FAIL and red data points outside the limit lines. • Nine dots (...) indicate that 90% of the data points achieved the specified compression level. • one ! indicates that 10% of the data points did not achieve compression. • Learn more about the Iteration Counter and annotation.

SMART Sweep Tips

- Compression from Linear Gain is the easiest compression method to understand and control in SMART Sweep. [Learn more.](#)
- If SMART Sweep requires more than twenty iterations, this is an indication that something is wrong. Try changing the Tolerance setting, Frequency Range, Start / Stop power range, IF bandwidth, or [Dwell Time](#).
- If the number of iterations required to achieve the desired compression level changes significantly from one set of measurements to the next, this could be due to other effects, such as heating. Try increasing the dwell time or using a [wideband pulse](#) measurement configuration.

- If the DUT should not be significantly overdriven into compression, or the changes in the input power should be limited, use [Safe Sweep](#) mode with Deviation from Linear Gain compression method.

Single Frequency Macros

Note: Beginning with PNA rev. A.09.00, the [Compression Analysis](#) feature provides an easier method of viewing a traditional power sweep at a single frequency than the GCA macros. However, the Macros are still maintained on the PNA hard drive.

The macros perform a single power sweep on the DUT using a standard channel with corresponding stimulus settings. The macro can show measurement differences from the compression analysis traces due to bias/thermal/settling effects of the DUT. So, the macro can help confirm a DUT is exhibiting some type of settling behavior which will need to be handled in some way.

Also, the macro is a great GCA programming example.

With a 2D sweep (NOT SMART Sweep) a script that is stored on the PNA hard drive automatically creates a traditional power sweep measurement in a standard channel using the same stimulus setting as the GCA channel. Use a marker in the GCA channel to specify the frequency for the measurement.

The script has two modes of operation:

1. **View Mode** displays all of the previous 2D sweep data at that frequency.
2. **Measure Mode** performs a new measurement at that frequency.

Both modes create a new S-Parameter channel using the same stimulus settings as the GCA channel, including port power, attenuator, IF Bandwidth, and dwell settings. The new channel does not support calibration or pulse characteristics.

To see noise on a measurement, use the **Measure** macro in continuous sweep. Adjust the IFBW and averaging until the noise versus sweep speed meets your needs.

To see other effects of your DUT at a specific frequency, use the **View** macro and the **Measure** macro with 2D sweep mode. Both macros present data using a standard channel. The View macro shows 2D data at a specific frequency, while the Measure macro shows freshly-measured data at the same frequency. Ideally, the data from these two would be identical. However, changes in your DUT behavior due to heating or other effects can cause these to be different. If significant differences exist, try:

- Using the 2D Frequency per Power setting rather than Power per Frequency
- Adjusting the dwell time
- Adjusting IFBW

- Use a [wideband pulse configuration](#)

How to setup the Macros

Each macro must be setup separately.

1. Press **Macro**, then **Macro Setup**.
2. Select a blank line, then click **Edit**.
3. In **Macro Title**, type a short description such as Meas GCA or View GCA.
4. Click **Browse**, then navigate to C:/Program Files/Agilent/Network Analyzer/Applications/GCA/GCA.vbs
5. In Macro run string parameters:
 1. Type **M** for the Measure macro or **V** for View macro.
 2. Optional: Supply the following additional parameters in any order:
 - To run the program from a remote computer, specify the full computer name of the PNA .
 - Channel in which to create the measurement. If not specified, Measure is created in Ch30 and View is created in Ch31.
 - Example: Run string parameters for the Measure macro run from a remote computer in Channel 5.----
M MyPNA 5.
6. Click **OK**.

How to run the Macros

On a GCA channel:

1. Create a 2D sweep. Either Power per Freq or Freq per Power. Both macros always create a power sweep at the frequency of interest.
2. Create a Compln trace.
3. On the Compln trace, right-click and select **Add Marker**. Drag the marker to the frequency of interest.
4. Press **Macro**, then select either by the short description your provided in Step 3.

Last Modified:

11-Sep-2013	Updated compression tab diag
7-May-2013	Removed narrowband pulse support
30-Aug-2011	Edited for 2D sweep data variance
24-Jun-2011	Removed point and sweep mode limitation
27-Apr-2011	Removed Copy Channels limitation
1-Apr-2011	Removed acquisition limitations
17-Mar-2010	Added Max power and pulse support - removed some fixturing limitations
19-Nov-2009	Added X-axis point spacing (9.1)
13-Aug-2009	Added Compression Analysis
30-Mar-2009	Added ADC measurements
23-Feb-2009	Added link to compression marker
1-Dec-2008	Clarified SMART - Linear compression
9-May-2008	Edited for data save, backoff, and XY method
23-Aug-2007	New topic

Gain Compression for Converters (GCX)

Gain Compression is measured on Converters in the same manner as it is measured in Amplifiers. Also, the Mixer/Converter setup is very similar to that of [SMC](#) (Scalar Mixer Converter application).

In this topic (unique for GCX):

- [Requirements and Limitations](#)
- [Using GCX](#)
- [Create a GCX Measurement](#)
- [Valid Mixer Configuration / Sweep Type Combinations](#)
- [Measurement Parameters offered in GCX](#)
- [GCX Calibration](#)

The following **Gain Compression for Amplifiers** information is relevant for learning about GCX:

- [Gain Compression Concepts](#)
- [Understanding the GCA Displayed Traces](#)
- [Compression Methods](#)
- [Acquisition Modes](#)
- [Compression Analysis](#)
- [Saving GCA Data](#)
- [GCA Measurement Tips](#)

The following dialog setup tabs are shared with other applications:

- [Frequency tab](#) (GCA topic)
- [Power tab](#) (GCA topic)
- [Compression tab](#) (GCA topic)
 - [Safe Sweep Mode dialog](#) (GCA topic)
- [Mixer Frequency tab](#) (separate topic)
- [Mixer Setup tab](#) (separate topic)

- [Mixer \(LO\) Power tab](#) (separate topic)

Other Applications Topics

Requirements and Limitations

GCX requires Option 086 (Gain Compression) and [FCA](#) (either Option 082 or 083).

Limitations:

- Number of points limited to 20,001 for two-dimensional acquisitions, 10,000 points for SMART Sweep.
- Linear and CW sweep ONLY - No Power, Log, or Segment sweep. [Learn more.](#)
- GCX does NOT provide any built-in image rejection techniques. You should provide image rejection hardware if necessary.
- Stepped sweep mode only.
- Does NOT support Narrowband Pulse measurements using the integrated [Pulse setup dialogs](#).

The following PNA features are **NOT** available with Gain Compression on Converters:

- [ECal User Characterization](#)
- [Time Domain](#)
- [Balanced measurements](#)
- Save [Formatted Citifile](#) data.
- [Save SnP data.](#)
- [Interface Control](#)
- [Port extensions](#)
- [Some Fixturing Features](#)
- [External Test Set Control](#) (Option 551)
- [Integrated Narrowband](#) or [Narrowband Pulse App](#)
- Independent IFBW, Power Levels, or Sweep Time in a [segment table](#) is NOT supported.

Using GCX

The following is a general procedure for performing a GCX measurement. The challenge with GCX is

configuring a measurement that yields the true compression performance of YOUR DUT. This requires knowledge of the Gain Compression and Mixer settings, and knowledge of the DUT.

See specific dialog boxes below for details.

1. Disconnect the DUT if preset or default power levels may damage the PNA or DUT.
2. [Preset](#) the PNA, or configure a suitable [User Preset](#) that will be safe in case the DUT is connected.
3. Create a GCX channel. [Learn how](#). The default trace is SC21.
4. Start the GCX Setup dialog and configure the measurement settings based on the DUT, adapters, attenuators, booster amplifiers, and fixtures to be used in the measurement. To start the dialog, click **Stimulus**, then **Frequency**, then **GCX Setup**. [Learn about the setup dialogs](#).
5. Save the [instrument state](#) (optional).
6. Connect the DUT. Inspect the measurement to ensure the DUT is operating as expected.
7. Add GCX compression and mixer parameter traces. [Learn more](#).
8. Adjust the measurement settings to yield satisfactory compression results. [See GCA Measurement Tips](#).
9. Start and complete the [GCX Calibration wizard](#).

Create a GCX Measurement

1. On the PNA front panel, press **Meas** then **[Measurement Class]**
2. Select **Gain Compression Converters**, then either:
 - **OK** delete the existing measurement, or
 - **New Channel** to create the measurement in a new channel.
3. A default SC21 measurement is displayed. To select additional parameters to display, click **Trace/Chan**, then **New Trace**, then select a parameter from the list. [Learn more about GCX Parameters](#).

How to start the Gain Compression for Converters Setup dialog

To provide quicker access, use the Setup softkey. [Learn how.](#)

Using front-panel HARDKEY [softkey] buttons

1. Press **FREQ**
2. then [**Gain Compression Setup**]

Using a mouse with PNA Menus

1. Click **Stimulus**
2. then **Frequency**
3. then **GCX Setup**

Programming Commands

Valid Mixer Configuration / Sweep Type Combinations

The following are the **Valid Sweep Type / Mixer Configurations**.

Sweep Type	Input	LO	Output
Linear	Swept	Fixed	Swept
	Swept	Swept	Fixed
CW	Fixed	Fixed	Fixed
	Fixed	Swept	Swept

For determining a valid mixer configuration with 2 LOs, one Fixed LO and one Swept is equivalent to having a single-stage Swept LO.

If you create an invalid Sweep Type / Mixer Configuration, a red message appears like the following:

ERROR: Input range must be swept in Linear sweep mode

If this occurs, change the **Sweep Type** on the [Frequency tab](#).

[See other rules for configuring a mixer.](#)

GCX Measurement Parameters

Note: The following table assumes: DUT **Input** = PNA **port 1** and DUT **Output** = PNA **port 2**.

When the Port mapping is different, the parameters in GCX are updated accordingly. For example, with Input = port 2 and Output = port 1, then "CompIn12" would be displayed.

Parameter	Description
Mixer Parameters	
SC21	Linear Conversion Gain
SC12	Reverse Conversion Gain
S11	Input Match
S22	Output Match
Compression Parameters	
CompIn21	Input power at the compression point.
CompOut21	Output power at the compression point.
CompGain21	Gain at the compression point.
CompS11	Input Match at the compression point.
RefS21	Linear Gain value used to calculate the compression level. This is calculated differently depending on the compression method.
DeltaGain21	CompGain21 MINUS Linear Gain (in Log Mag format). This trace can be used to learn a lot about the DUT compression point. Learn more.
Unratioed - Absolute test port receiver measurements. Learn more.	
IPwr	Input power measured at DUT-IN @ Input frequency
OPwr	Output power measured at DUT-OUT @ Output frequency
RevIPwr	Input power measured at DUT-OUT @ Output frequency
RevOPwr	Output power measured at DUT-IN @ Input frequency
ADC Parameters - Learn more.	
AI1	Measured at the specified Linear Input level.
AI2	Measured at the specified Linear Input level.
CompAI1	AI1 at Compression
CompAI2	AI2 at Compression

GCX Setup Dialogs

All of the GCX Setup tabs are shared with other applications.

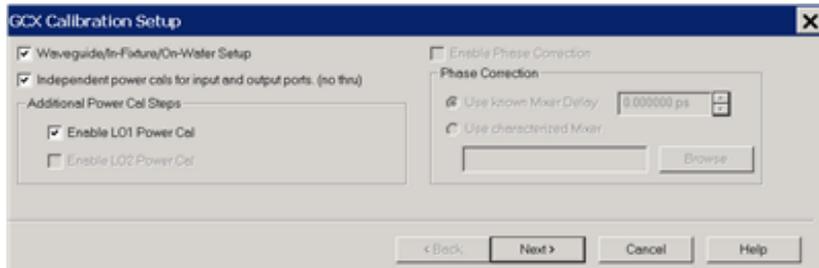
- [Frequency tab](#)
- [Power tab](#)
- [Compression tab](#)
 - [Safe Sweep Mode dialog](#)

- [Mixer Frequency tab](#)
- [Mixer Setup tab](#)
- [Mixer \(LO\) Power tab](#)

GCX Calibration

A GCX Cal is conceptually the same as a [Gain Compression Calibration](#). This includes the ability to perform or downgrade to an Enhanced Response Cal. [Learn how.](#)

The following Guided Cal Wizard pages are unique to GCX:



GCX Calibration Setup dialog box help

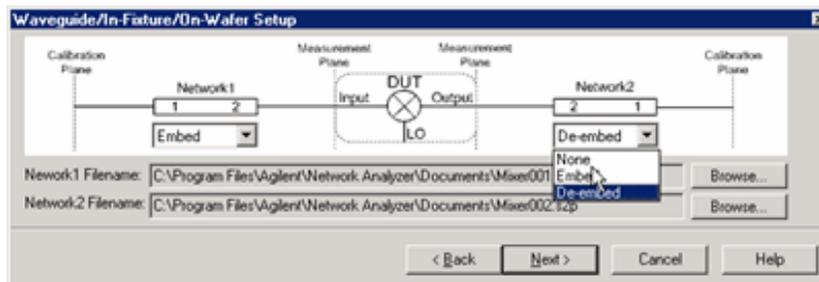
Waveguide/In-fixture/On-Wafer Setup Starts the following dialog box.

Independent power cals for input and output ports (no thru) Check if a Thru standard is NOT available. During the power cal, you will be prompted to connect the power sensor to the Input, then the Output port.

Additional Power Cal Steps

Enable LO1 / LO2 Power Cal Check when LO1 / LO2 is controlled (on the [Mixer Setup](#) tab) to perform a Power Cal on the source.

Note: Phase Correction is NOT allowed for GCX measurements.



Waveguide/In-fixture/On-Wafer Setup dialog box help

This dialog box appears ONLY if you checked the **Waveguide/In-fixture/On-Wafer Setup** box in the previous [Cal Setup](#) dialog.

Allows you to embed or de-embed circuit networks on the input and output of your mixer under test.

For Network1 (Input) and Network2 (Output) select **Embed**, **De-embed**, or **None**.

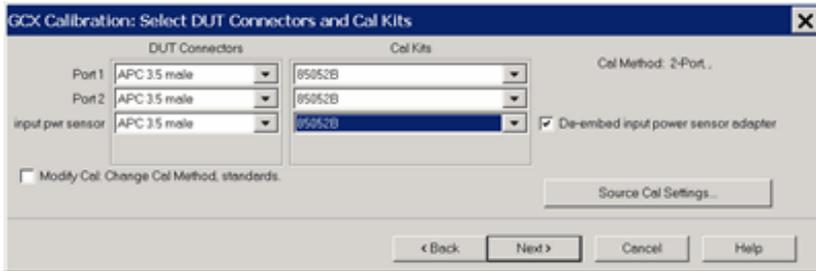
Browse Click to navigate to the .S2P file that models the network to embed or de-embed.

To Embed or De-embed

- **De-Embed** when there is a 2-port network that needs to be connected during the measurement, but it is NOT present during the calibration. An example might be when you do not have standards for a characterized test fixture, so you calibrate without the fixture, and make measurements with the DUT in the fixture. De-Embedding during the calibration extends the calibration reference plane to include the 2-port network.
- **Embed** when there is a 2-port network that to be disconnected during the measurement, but is present during the calibration. An example might be when a characterized adapter is required during the calibration but NOT present during measurements of the DUT. Embedding during the calibration retracts the calibration reference plane to exclude the 2-port network during the measurement.

Notes

- [Characterize Adaptor Macro](#) can be used to create the S2P file.
- The S2P file for Network1 (on the input of the mixer), must cover the Input frequency range. The S2P file for Network2 (on the output of the mixer), must cover the Output frequency range.
- The frequency range of the S2P file must be the same, or larger than, the frequency range of the measurement. If more frequencies are included in the file, and the data points do not exactly match those of the measurement, interpolation will be performed.
- As in the image on the dialog (above), in all cases:
 - Port 1 of each network is assumed to be connected to the PNA.
 - Port 2 of each network is assumed to be connected to the DUT.



Select DUT Connectors and Cal Kits dialog box help

Allows you to specify the connector type and Cal Kit for each DUT port.

Port n For each listed PNA port, specify the DUT connector type and gender, and the Cal Kit to use.

input pwr sensor Specify the connector type of the power sensor. Select **Ignored** to not compensate for the effects of the adapter that may be necessary to connect the power sensor to the input reference plane.

output pwr sensor Available when **Independent power cals for input and output ports** is checked on the [GCX Calibration Setup](#) dialog. Specify the connector type of the power sensor. Select **Ignored** to NOT compensate for the effects of an adapter that may be necessary to connect the power sensor to the output reference plane.

De-embed input power sensor adapter Check to measure, then remove the effects of the adapter that is used to connect the power sensor to the calibration reference plane.

Source Cal Settings Click to start the [Source Cal Settings](#) dialog. These settings allow you change ALL Source Cal and Power Meter settings.

Note: If your DUT connectors are:

- **Waveguide** Change the system impedance to 1 ohm before performing a calibration. See [Setting System Impedance](#).
- **Not listed** (male and female) Select **Type A** as the connector type. Type A requires a calibration kit file containing the electrical properties of the standards used for calibration (see [Calibration kits](#)).
- **Unspecified** (like a packaged device) Select **Type B** as the connector type. Type B requires a calibration kit file containing the electrical properties of the standards used for calibration (see [Calibration kits](#)).

Modify Cal Check, then click Next, to start the [Modify Frequency Cal dialog](#).

27-Apr-2011 Removed Copy Channels limitation
9-Aug-2010 New topic

Gain Compression Calibration

The GCA Calibration Wizard guides you through a calibration of GCA or GCX channel. The procedure is the same regardless of the Gain Compression Settings.

- A [Source Power Calibration](#) is performed first.
- Then, [your choice of a Full 2-port Cal or an Enhanced Response Cal](#).

See Also

[Gain Compression Application](#)

[Gain Compression for Converters](#)

[Calibration Programming commands](#)

How to start a GCA Calibration	
Using front-panel HARDKEY [softkey] buttons	Using a mouse with PNA Menus
<ol style="list-style-type: none">1. Press CAL2. then [Start Cal]3. then [Cal Wizard]	<ol style="list-style-type: none">1. Click Response2. then Cal Wizard

 **Programming Commands** 

Overview - GCA Source Power Cal

The GCA Calibration Wizard first performs a Source Power Cal. The GCA Source Power Cal is a little different from a standard Source Power Cal. Although GCA measurements are performed at many power levels, the GCA source power cal is performed at a [single power level](#) over the specified frequency span of your GCA measurement. The required source correction from that single power level is applied to ALL power levels. This method ensures that the 'absolute' power level being applied to the DUT is within the PNA-X source power linearity specification.

Although it is important for GCA to be able to **set** the absolute power level to the DUT, it is **MOST** important to be able to exactly **measure** the actual incident power. Therefore, during the GCA Source Power Cal, a receiver calibration is applied to the port 1 reference receiver, and indirectly to both test port receivers during the S-parameter calibration, correcting for impedance mismatch between the

power meter and the PNA source, and the DUT and the PNA source.

Although the cal process is also at a single power level, the dynamic accuracy of the PNA-X receivers is typically about +/- .05 dB, which is comparable to the accuracy of Agilent's best power sensors. This allows GCA to **very accurately** measure and report ALL power levels that are actually applied to the DUT.

Full 2-port or Enhanced Response (ER) Cal

By default, a full 2-port calibration is performed as part of a GCA and GCX calibration. However, you can change to an [Enhanced Response](#) Cal. The following issues may help you decide between these two Cal types:

- **Accuracy** A full 2-port correction is more accurate than ER when GCA measures linear gain. However, for non-linear measurements, ER yields identical compression values as a full 2-port cal, so this may not be a significant factor.
- **Measurement speed** An ER correction only requires measurements in the forward direction. The reverse parameters (usually S22 and S12) are not measured unless requested. With a full 2-port cal applied, all four S-parameters are measured, which requires an additional reverse sweep. [Learn more.](#)
- **Ease** A full 2-port cal is easiest with an ECal module. An ER Cal requires a [Defined Thru](#) or a [Flush Thru Cal](#) method. If these are possible, then an ER cal is easiest when using a mechanical Cal Kit.
- **High power** The test port damage level of a standard PNA-X is +30 dBm. Therefore, external attenuation may be required on the output of high power amplifiers, which degrades calibration accuracy for reverse (full 2-port) measurements. In addition, the external attenuation improves the DUT output / load match error, which allows a better uncorrected response and makes an Enhanced Response Cal the better choice.
- **DUT limitations** With an ER Cal applied, reverse measurements on the DUT are not performed unless requested.

How to select Enhanced Response Cal

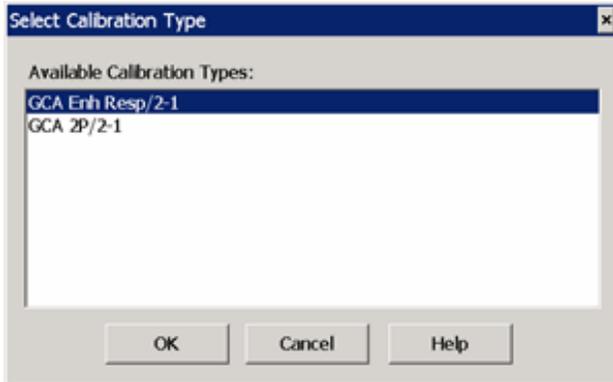
At the [Select DUT Connectors](#) page of the GCA Cal Wizard:

1. Check **Modify Cal**, then click **Next**.
2. A [Defined Thru](#) or a [Flush Thru Cal](#) method must be selected.
3. Click **Cal Type/Std**
4. Under Calibration type, select **EnhResp** (2 <= 1 refers to the receive port 2 and source port 1).

Downgrade a Full 2-port Cal to Enhanced Response Cal if you prefer to perform a Full 2-port cal, but not perform reverse sweeps on the DUT.

To change the correction on the channel from Full 2-port to Enhanced Response:

1. Press **CAL**
2. then **[Manage Cal]**
3. then **[Correction Methods]**
4. Select **GCA EnhResp**, then **OK**



GCA Cal Wizard



Select DUT Connectors and Cal Kits - GCA Cal dialog box help

This is a [standard Cal Wizard](#) page except for the following:

Power Sensor Specify the connector type and gender of the power sensor. When the power sensor connector is not the same type and gender as the DUT Port 1 connector, then an adapter is required to connect the power sensor to the port 1 reference plane during the Source Power Cal. An extra 1-port cal is performed to measure and correct for the adapter. No characterization S2P files are required.

- Select **Ignored** (at the bottom of the DUT Connectors list) to NOT compensate for the adapter.
- Select the Cal Kit that will be used for that process.

Modify Cal Check, then click **Next**, to Modify Cal (Standards AND Thru Method).

Source Cal Settings Click to launch the [Source Cal Settings](#) dialog.

[Learn more about GCA Source Power Calibration](#)



Gain Compression Calibration Step 1 dialog box help

Power Level at which to perform the Source Power Cal.

It is usually best to perform the Source Power Cal at 0 dBm because the power sensor is calibrated at that level.

However, if the Gain Compression measurement is performed entirely below or above 0 dBm, then perform the Source Power Cal at the **Stop** power which probably has the lowest level of measurement noise.

[Learn more about GCA Source Power Calibration](#)

The remaining Gain Compression Cal dialogs are the same as the standard [SmartCal dialogs](#).

Return to [Gain Compression Application](#).

Last Modified:

6-Feb-2012	Slight edit to first paragraph
17-Oct-2011	Clarified adapter selection
27-Sep-2010	Updated for GCX
26-Nov-2007	MX New topic

Integrated Pulse Measurements

The Pulse Setup dialogs shown in this topic are now integrated in the PNA firmware and are available with Opt 008 or Opt H08.

Previously, setup was performed with the [Narrowband](#) or [Wideband](#) pulse programs. With the appropriate hardware options (Opt 021, 022, 025) these commands are still available without Opt 008 or H08. [Learn more about PNA Options.](#)

Beginning with A.09.50, external pulse generators can be used along with the PNA internal pulse generators. [Learn more.](#)

In this topic

- [Pulse Setup](#)
- [Pulse Generator Setup](#)
- [Pulse Trigger Tab](#)
- [Pulse Gens and IF Block Diagram](#)
- [Calibration in Pulse](#)

See Also (separate topics)

- [Configure and Use External Pulse Generators](#)
- [IF Path Configuration](#)
- [Programming commands](#)
- [Narrowband Pulsed Application](#) (Opt H08)
- [WB Pulsed App](#)
- [See Swept IMD note](#) regarding IF Filter setting

App Note: [Active-Device Characterization in Pulsed Operation Using the PNA-X \(1408-21\)](#)

How to start the Pulse Setup dialog

To provide quicker access, use the Setup softkey. [Learn how.](#)

Using front-panel HARDKEY [softkey] buttons

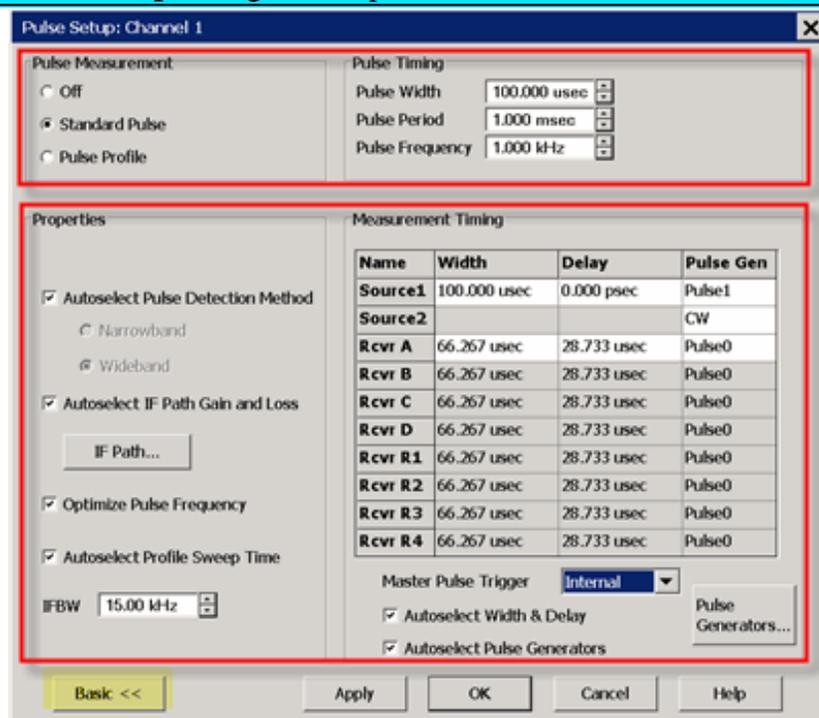
1. Press **SWEEP**
2. then **[Pulse Setup]**

Using a mouse with PNA Menus

1. Click **Stimulus**
2. then **Sweep**
3. then **Pulse Setup**

Programming Commands

Pulse Setup dialog box help



The Basic (upper box) controls allow simple pulse measurements using the default (Autoselect) settings in the Advanced section (lower box) of the dialog.

Pulsed measurements are performed in a Standard channel. [See Measurement Class.](#) However, several PNA measurement settings are controlled by the Pulse setup, such as sweep type, number of points, and so forth.

Pulse Measurement

Off - Source and Receivers are NOT pulsed

Standard Pulse - With pulsed RF, the PNA can be configured to sweep in frequency, power sweep, and CW time.

- To make 'Point-in-Pulse' measurements, narrow the receiver pulse width and enter delay.
- To make 'Pulse-to-Pulse' measurements, disable **Autoselect Pulse Detection Method**, then select **Wideband**. In the PNA menu, select **Stimulus**, then **Sweep**, then **Sweep type = CW**.

Pulse Profile - Pulse profile measurements provides a time domain (CW frequency) view of the pulse envelope. Profiling is performed using a measurement technique that "walks" a narrow receiver "snapshot" across the width of the pulse. This is analogous to using a camera to take many small snapshots of a wide image, then piecing them together to form a single, panoramic view.



Pulse Profile measurement using default settings and R1 receiver.

- Pulse Profiling can be performed using ratioed or unratioed measurements. You can preview the pulse on port 1 by using an R1 receiver measurement.
- Pulse Profiling is performed at a single CW frequency in either Narrowband or Wideband mode.
- To select the CW Frequency, click **Stimulus**, then **Sweep Type**.
- In Narrowband mode, the delay increment value, which is responsible for "walking" the receiver snapshot across the pulse, is selected by the PNA and is accessible only with a [programming command](#).
- In Wideband mode, the receiver is walked across the pulse by making a sequence of closely-spaced measurements in real-time.

Pulse Timing

Pulse Width - Sets the width of the source pulse. See [measurement timing](#) to learn how to control the receiver width and delay.

Pulse Period The time to make one complete pulse.

Pulse Frequency (PRF) The reciprocal of Period (1/ Period). See [Internal Pulse Generators](#) to learn more.

By default, these settings configure Pulse Gen 1 to drive Source Modulators 1 and 2. This can be changed from the Advanced Settings [Pulse Generator Setup](#) dialog.

----- Advanced Settings -----

The following settings allow maximum control of a Pulse measurement.

Note: When the "Auto" check boxes are cleared, it is possible to configure settings to make an invalid measurement.

Properties

Autoselect pulse detection method - check to automatically switch between Narrowband and Wideband based on the Pulse Width.

In Standard Pulse:

- **Wideband** - used when the (source) Pulse Width is WIDER than the fastest receiver acquisition time. This allows the receiver to measure all pulse ON time - no pulse OFF time. The PNA will select Wideband whenever possible.
- **Narrowband** - used when the (source) Pulse Width is NARROWER than the fastest receiver acquisition time (267 ns). This measurement requires a spectral nulling technique to measure the pulse response through the DUT.

In Pulse Profile:

- **Wideband** - used when the (source) Pulse Width is greater than 1.600 us. This allows the receiver make several sequential measurements to measure the entire pulse.
- **Narrowband** - used when the (source) Pulse Width is less than or equal to 1.600 us.

Autoselect IF Path Gain and Loss - For future use.

IF Path - Click to launch the [IF Path dialog](#).

Optimize Pulse Frequency - Automatically selects the Pulse Frequency and Pulse Period.

- In Narrowband, the pulse frequency is adjusted slightly to get the best spectral-nulling filtering possible.
- In Wideband, this checkbox is ignored.

Autoselect Profile Sweep Time - In Pulse Profile mode, adjusts the default X-axis start time to zero and the stop time double the Pulse Width. This allows you to see one complete pulse. If unchecked, the Sweep Time will not be changed.

To adjust the X-axis manually, click **OK** to close the dialog. Then click **Stimulus**, then **Sweep**, then **Sweep Time**, then change the **Start Time** and **Stop Time**.

IFBW - Select the IFBW for the measurement.

- In Narrowband, an IFBW as close as possible to the entered value will be used.
- In Wideband, this setting determines the receiver acquisition time - approximately 1/ IFBW.

Measurement Timing

Source1 and **Source2** - Used as RF Source Modulation Drive.

- **Width** - source pulse width.
- **Delay** - source pulse delay relative to the pulse generator clock.
- **Pulse Gen** - Pulse generator used to modulate the source. Select **CW** to have NO source modulation.

The receiver settings in this table change depending on whether the PNA is in Narrowband or Wideband mode.

- In Narrowband, for each IF receiver path, configure the Pulse Width, Delay, and Pulse Generator to be used to drive the receiver gate.
- In Wideband, all receiver paths are the same.

Master Pulse Trigger Refer to [Block Diagram](#) Choose from:

- **Internal** - Default setting. Pulse0 generator is used to trigger the ADC to make pulse measurements.
- **External** - An external pulse generator is selected but not controlled by the PNA. Use this setting to make manual pulse measurements.
- **<External Pulse Gen name>** - Available when a 81110A is [configured as an External Device](#) and **Master Mode** is checked on the [pulse generator properties dialog](#). See how to make this setting using [SCPI](#) and [COM](#).

Autoselect Width and Delay - When checked, for Wideband mode and Pulse0, the default setting for the receiver is adjusted to approximately 75% of the source pulse width, with 20%

delay. This leaves approximately 5% of the source pulse ON after acquisition is complete. This pulse width is further adjusted after an IFBW is selected.

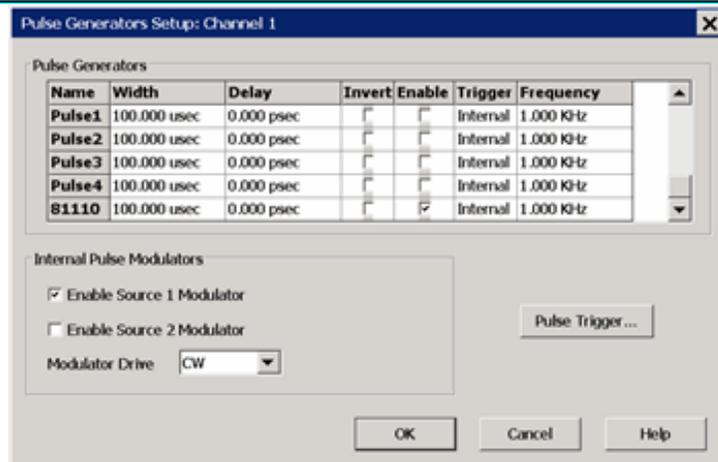
When checked for Narrowband mode and Pulse Gen = CW, then the delay and width have no values. When a Pulse Gen is selected, then Delay and Width matches the RF Source.

Autoselect Pulse Generators - When checked:

- Pulse1 is selected for Modulator Drive.
- Pulse2 is CW (OFF).
- For Wideband, Pulse0 is selected to gate the ADC.
- For Narrowband, Pulse2 is selected.

Pulse Generators Click to launch the [Pulse Generators Setup](#) dialog.

Pulse Generators Setup dialog box help



This dialog is available with Option 025 (pulse generators).

To see this dialog, press **Pulse Gen Setup** on the [Pulse Setup dialog](#).

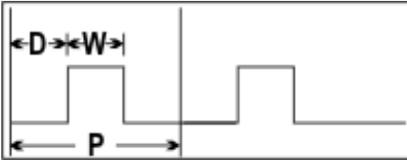
Pulse Generators

Configure the Pulse Generators to be used for your measurement.

When configured, external pulse generators appear in this list. [Learn more](#).

Pulse0 triggers the PNA receivers when Enabled. The Width can NOT be configured. Delay defines the delay between the pulse clock and when the receiver begins to make a measurement. [Learn](#)

[more.](#)



- **D** = Delay; the time before each pulse begins
- **W** = Width; the time the pulse is ON
- Duty Cycle = W/P
- **P** = Period; one complete pulse cycle
- Pulse Frequency (PRF) = $1 / \text{Period}$

Important: If **D + W** is greater than **P**, then undefined PNA behavior results. There is NO error message or warning.

Invert Check to cause the pulse ON time to be active low and OFF be active high.

Enable Check to enable individual pulse generators.

Trigger Choose from: (When ONE of these is changed, they ALL change. The internal Pulse Generators can NOT be triggered individually).

- Internal - Pulse generators are triggered by the internal pulse clock.
- External - Pulse generators are triggered by an external pulse generator.

Frequency - Set the pulse frequency of each generator.

- Pulse Frequency (PRF) = $1 / \text{Period}$
- **P** = Period; one complete pulse cycle

[Learn more about the Pulse Generators.](#)

Internal Pulse Modulators

Check to enable one or both internal source modulators.

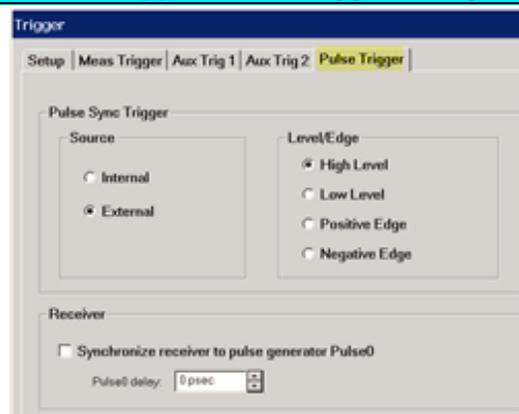
These are switches 8 and 9 in the [Block Diagram](#)

Important: When internally modulating the sources, [source leveling is automatically set to Open-loop.](#)

Modulator Drive Choose the pulse generator to modulate the specified source. Choose from **CW** (NO pulse), Pulse 1, 2, 3, 4, External. This is switch 7 [Block Diagram](#).

Pulse Trigger.. Click to start the [Pulse Trigger dialog](#).

Pulse Trigger Tab - Trigger dialog box help



To see this dialog, press **Pulse Trigger** on the [Pulse Generator Setup dialog](#) or select **Stimulus**, then **Trigger** from the PNA Menu.

Pulse Sync Trigger - Source

Select **Internal** or **External** to provide sync capability for the internal pulse generators.

- **Internal** - The pulse generator is internally triggered and puts out a periodic pulse train with a period defined by the [Pulse Generator Setup dialog](#).
- **External** - The period is ignored and the internal pulse generator puts out one set of pulses (P0-P4) per external trigger. All five pulse outputs have unique delay and pulse width settings.

The external trigger input is on the [Pulse I/O connector](#) pin 7 (PulseSyncIn). The PulseSyncIn line provides a configurable trigger signal into the Pulse Generators. If a level trigger is still valid when the first pulse set is finished, another set will be generated. Only one set of pulses is emitted when edge triggering is used. The length of time that it takes to emit one set of pulses is the end time of the last enabled pulse (largest of width + delay of all the pulses P0-P4).

Level/Edge

Sets the edge or level of the trigger signal to which the internal pulse generators will respond when being externally triggered at the PulseSyncIn pin.

Positive = rising edge; Negative = falling edge.

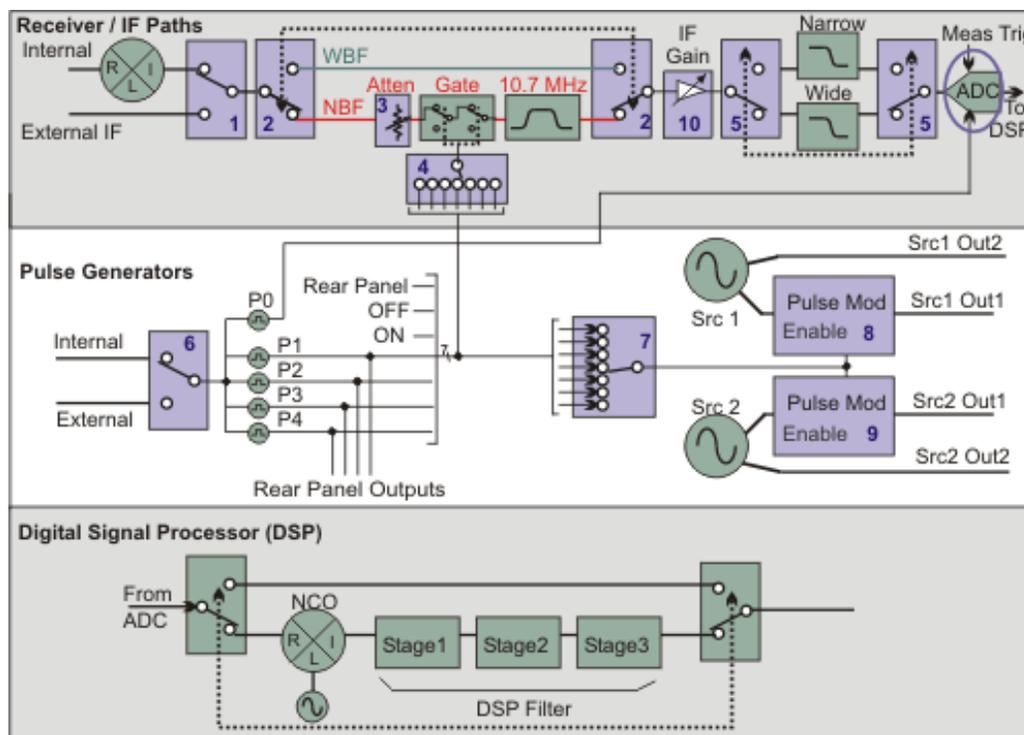
These selections are available ONLY with DSP version: 4.0 FPGA: 34 or higher. [Learn more.](#)
Otherwise, the pulse generators respond only to positive, level input trigger signals.

Receiver

Synchronize receiver to pulse generator Pulse0 - Check to enable Pulse Gen 0 (P0) which is used to gate the ADC for wideband receiver measurements. [Learn more about Pulse0.](#)

Receiver delay - Set the amount of time to wait before triggering the ADC to begin acquisition.

Pulse Trigger Block Diagram



[See complete description at IF Path Configuration](#)

See also [Using External Pulse Generators](#)

Using External Pulse Generators

[Setup the External Pulse Generator as an External Device.](#)

Calibration in Pulse Mode

To perform a calibration in pulse mode (option H08 / 008), first configure and apply the pulse parameters (PRF, Pulse Width, Delays, IF gating, and so forth) **before** calibrating the system. This will ensure the PNA is configured properly during the calibration and measurement.

Last Modified:

19-Dec-2013	Added Pulse-to-Pulse
11-Apr-2010	MX New topic (A.09.20)

iTMSA (Integrated True Mode Stimulus Application)

This application is integrated into the PNA firmware as **Option 460**. A previous version of TMSA is run as a Macro on the PNA.

- [Features, Requirements, and Limitations](#)
- [True Mode Stimulus Concepts](#)
- [Using iTMSA](#)
 - [Create a Measurement](#)
 - [Set Power Level](#)
 - [Calibrate](#)
 - [Ratioed Receiver Measurements](#)
 - [Differential Input and Output Power Measurement](#)

Other PNA Applications

Important Note

When the first iTMSA measurement is created, an IFMUX Cal is performed, which takes a couple of minutes.

When finished, **Exit** then **restart** the PNA App.

If an IF board is replaced, this Cal should be performed again. To make this happen, delete `c:\users\public\network analyzer\IFMUX.txt`. When the next iTMSA measurement is created, the file will be recreated with new data.

Features, Requirements, and Limitations

Features

- A seamless extension of existing PNA Balanced Measurement, but with True Mode stimulus.
- True Mode Stimulus measurements are performed in a standard S-parameter channel.

Requirements

- [4-port PNA-X](#) (opt 400) with **Opt 460** (software option only); [must be enabled](#).

Limitations with iTMSA

- Direct Receiver Access configuration of the front-panel loops is NOT allowed. However, inserting amplifiers or other components into the standard source path configuration at the front-panel loops IS allowed.
- Reversing the port couplers is NOT recommended due to the increased noise on the [initial sweep](#) for match correction. Good reflection measurements on all four ports is necessary to ensure adequate accuracy.

The following standard PNA features are **NOT** available with iTMSA measurements:

- [External Multiport Test Set Control](#) (Option 551)
- [Interface Control](#)
- Time Domain Pulse measurements in [Wideband Pulse](#) are NOT supported.
- [Segment Sweep](#)
- [Frequency Offset Measurements](#) (opt 080)
- [mmWave Measurements](#) are allowed but with differential and S-Parameter measurements only.

True Mode Stimulus Concepts

A balanced device is designed to receive input simultaneously across two ports. Standard PNA Balanced measurements apply stimulus to one port at a time, measures the responses, and calculates the theoretical balanced responses. [Learn more about balanced measurements](#).

True Mode Stimulus uses two PNA sources to apply either truly differential (180 degree out-of-phase) or truly common (in-phase) signals across the input of a balanced device. PNA receivers measure the single-ended response at the output of the device and calculate the balanced response.

When operating in non-linear regions, a device may respond differently to single-ended stimulus than to True Mode Stimulus. Thus, True Mode Stimulus capability allows you to understand when, and if, True Mode Stimulus is required.

For more detailed information on this measurement technique, see the following white papers (internet connection required):

- [New Methods & Non-Linear Measurements for Active Differential Devices](#)
- [New Measurement Results and Models for Non-linear Differential Amplifier Characterization](#)

How iTMSA Works - Overview

The following is an overview of how iTMSA gathers and displays True Mode Stimulus data:

1. **Initial sweep** is performed to gather the initial phase deviation between the two sources at the reference plane.
2. **Measurement** The phase deviation between the two sources is set to the required value and the response is measured. For a differential sweep, the phase deviation of the two sources is set to 180° at the reference plane. For a common sweep, the phase deviation is set to 0° at the reference plane.

A forward direction measurement requires 2 sweeps for differential and 2 sweeps for common. A reverse direction measurement requires the same. Therefore, a complete measurement in both directions requires 8 sweeps. The iTMSA traces are updated when all of the necessary data is gathered in each direction.

iTMSA computes the raw S parameters with the following matrix:

$$\begin{bmatrix} \frac{b_1}{a_1} d1 & \frac{b_1}{a_2} d2 & \frac{b_1}{a_1} c1 & \frac{b_1}{a_2} c2 \\ \frac{b_2}{a_1} d1 & \frac{b_2}{a_2} d2 & \frac{b_2}{a_1} c1 & \frac{b_2}{a_2} c2 \\ \frac{b_3}{a_1} d1 & \frac{b_3}{a_2} d2 & \frac{b_3}{a_1} c1 & \frac{b_3}{a_2} c2 \\ \frac{b_4}{a_1} d1 & \frac{b_4}{a_2} d2 & \frac{b_4}{a_1} c1 & \frac{b_4}{a_2} c2 \end{bmatrix} \cdot \begin{bmatrix} \frac{a_1}{a_1} d1 & 0 & \frac{a_1}{a_1} c1 & 0 \\ 0 & \frac{a_2}{a_2} d2 & 0 & \frac{a_2}{a_2} c2 \\ \frac{a_3}{a_1} d1 & 0 & \frac{a_3}{a_1} c1 & 0 \\ 0 & \frac{a_4}{a_2} d2 & 0 & \frac{a_4}{a_2} c2 \end{bmatrix}^{-1} = \begin{bmatrix} S_{11} & S_{12} & S_{13} & S_{14} \\ S_{21} & S_{22} & S_{23} & S_{24} \\ S_{31} & S_{32} & S_{33} & S_{34} \\ S_{41} & S_{42} & S_{43} & S_{44} \end{bmatrix}$$

Legend:

- d1**: differential stimulus in balanced port 1.
- d2**: differential stimulus in balanced port 2
- c1**: common stimulus in balanced port 1.
- c2**: common stimulus in balanced port 2.

Using iTMSA

A Standard (default) S-Parameter channel is used for iTMSA measurements.

How to create an iTMSA trace:

1. Preset the PNA
2. Then do the following:

Using front-panel HARDKEY [softkey] buttons

1. Press **MEAS**
2. then [**Balanced Parameters**]
3. then press the **Change** button

Using a mouse with PNA Menus

1. click **Response**
2. then **Measure**
3. then **Balanced Parameters**
4. then press the **Change** button

Programming Commands

Balanced DUT Topology / Logical Port Mapping (with iTMSA Option 460) dialog box

Balanced DUT Topology / Logical Port Mapping

Topology: BAL-BAL

Port 1, Port 2, Port 3, Port 4

Balanced Port 1, DUT, Balanced Port 2

Stimulus

- Single Ended
- True Mode
- Forward True Mode
- Reverse True Mode

Balanced Port Offset

Select Port: Port 3

Phase Offset: 0.000 deg

Offset as fixture

Power Offset: 0.000 dB

Offset as fixture

Sweep(CW Time Only)

Enable Phase Sweep

Sweep Phase: Port 3

Start Phase: 0.000 deg

Stop Phase: 360.000 deg

Offset as fixture

OK Cancel Help

[Learn about iTMSA.](#)

Topology

Only the following topologies and port mappings are available in True Mode Stimulus:

- **BAL-BAL** DUT has two balanced ports. The PNA ports are restricted to the following:
 - Balanced Port 1 = PNA ports 1 and 3
 - Balanced Port 2 = PNA ports 2 and 4
- **SE-BAL** DUT has one single-ended port and one balanced port. For a **BAL-SE** (balanced - single-ended) DUT, select this topology, and then select **Reverse True Mode**. The PNA ports are restricted to the following:
 - Single-ended Port 1 = PNA port 1.
 - Balanced Port 2 = PNA ports 2 and 4.

Stimulus

Single Ended Standard PNA mode. Each DUT port receives stimulus individually.

True Mode Each balanced port receives true balanced stimulus in both forward and reverse directions.

Forward True Mode Available ONLY in BAL- BAL topology. Only Balanced port 1 receives true balanced stimulus.

Reverse True Mode Available ONLY in SE- BAL topology. Only Balanced port 2 receives true balanced stimulus.

Balanced Port Offset

Select Port Select the physical PNA port to receive a phase or power offset. Only port 3 and port 4 are available for Phase or Power offset.

Phase Offset Specify offset within +/- 180° for the balanced INPUT port. This is in addition to the standard offsets, which are 180° offset for the differential stimulus sweeps, and 0° for the common stimulus sweeps.

Offset (phase) as Fixture

When unchecked, output calculations are performed and displayed as though there is only the standard offset. Although additional Phase Offset is applied to the stimulus, it is ignored in the calculations of balanced differential and common mode output signals.

When checked, output calculations are performed and displayed using the actual phase offset that is applied to the DUT. Use this setting to compensate for a component or fixture that may

present a phase delay before the DUT.

Power Offset Specify power offset. Range is +/- 2 dB. This is in addition to the power level that is specified using the [Power and Attenuators dialog](#). Offset Power is NOT reflected on the power dialog nor on the X-Axis during a power sweep.

Offset (power) as Fixture

When unchecked, output calculations are performed and displayed as though there is no stimulus power offset.

When checked, output calculations are performed and displayed using the power offset that is applied to the DUT. Use this setting to compensate for a component or fixture that may present a magnitude loss before the DUT.

Phase Sweep

- Available only when [Sweep type](#) = CW.
- Only PNA port 3 **OR** port 4 are eligible for Phase sweep.
- Available **ONLY** for balanced pairs.

The phase of the selected port (3 or 4) is swept relative to the phase of the other port (1 or 2) in the balanced pair.

Phase Sweep is similar to phase offset, except that for each data point, the phase 'offset' is incremented.

For example, with the topology that is selected in the above image: (Bal-Bal) Logical port 1 = PNA ports 1 and 3. For a phase sweep with 7 data points, from 0° to 180°, the phase difference between port 1 and port 3 increments 30° with each data point:

Data point	1	2	3	4	5	6	7
Delta phase	0°	30°	60°	90°	120°	150°	180°

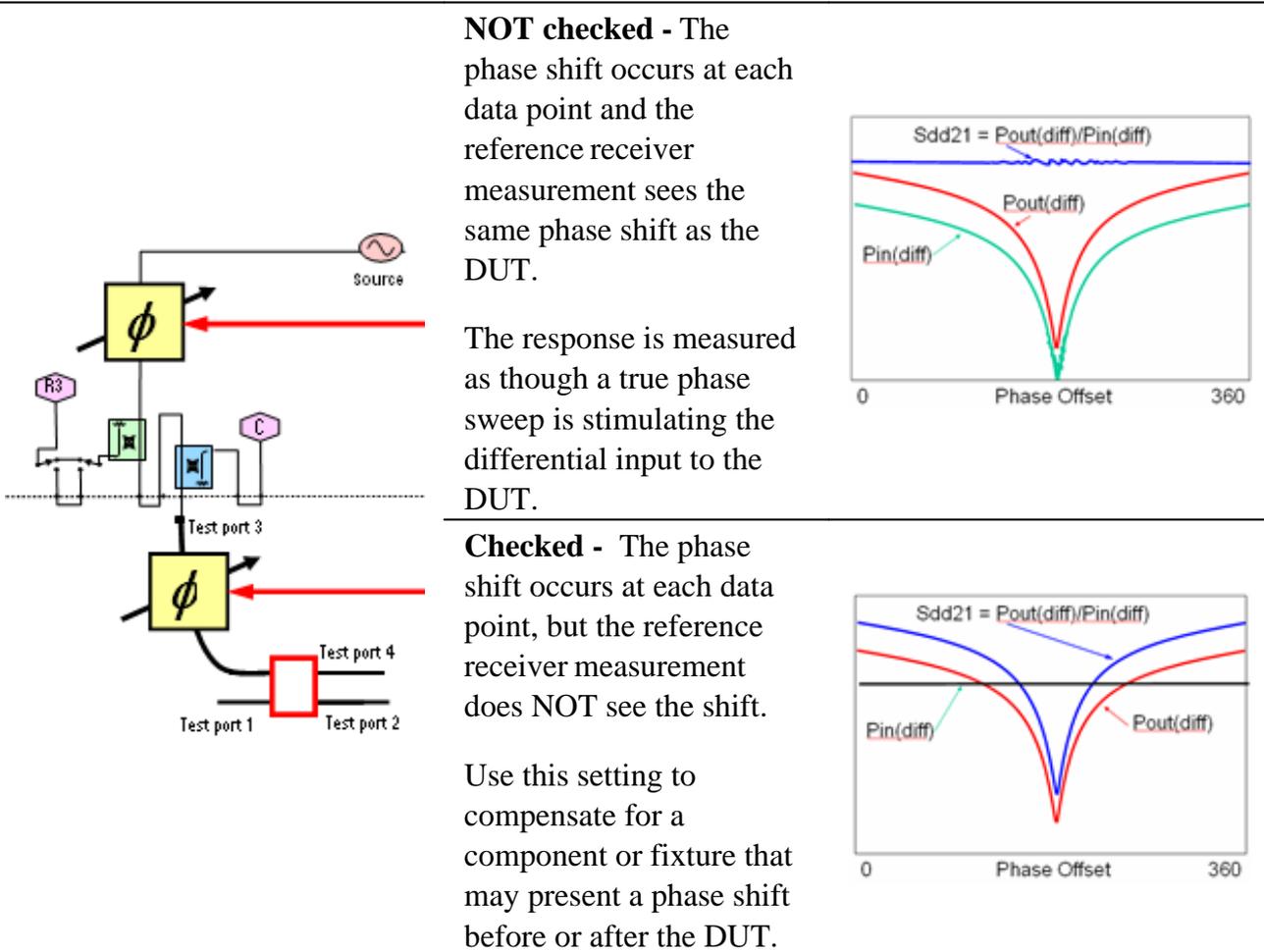
Enable Phase Sweep Check to enable phase sweep for the specified port.

Sweep Phase Choose EITHER PNA Port 3 OR Port 4 to sweep. Only ONE port can be phase swept.

Start / Stop Phase Enter phase values in degrees. Each sweep will start and stop at these settings.

Offset (phase) as Fixture

In the following image, the PNA phase sweep is shown as a phase shifter that is "virtually" located **EITHER** before the reference receiver (top) **OR** after the reference receiver (bottom) - **NOT BOTH**. This image is useful as a model **ONLY**.



One use for Phase Sweep

Do the following to find the best operating point:

1. Phase sweep port 3, from 0° to 180°. Note the peak of Sdd21. This finds the phase offset of the input differential port.
2. Set port 3 **fixed offset** to the phase that has the highest Sdd21.
3. Then, phase sweep port 4 (balanced port 2) to find the phase offset of the output differential port.
4. Then set the port 4 to fixed phase offset found.

This measures the best possible case Sdd21 for the DUT.

Power and Attenuators Settings

Set stimulus power levels using the standard PNA Power and Attenuators dialog.

To set power, press **STIMULUS**, then **[Power and Attenuators]**

Name	State	Port Power	Start Power	Stop Power	Auto Rang	Atten. Contr	Leveling Mod
Bal Port 1	Auto	-2.00 dBm	-7.00 dBm	3.00 dBm	<input checked="" type="checkbox"/>	0 dB	Internal
Bal Port 2	Auto	-2.00 dBm	-7.00 dBm	3.00 dBm	<input checked="" type="checkbox"/>	0 dB	Internal
Port 1	Auto	-5.00 dBm	-10.00 dBm	0.00 dBm	<input type="checkbox"/>	0 dB	Internal
Port 2	Auto	-5.00 dBm	-10.00 dBm	0.00 dBm	<input type="checkbox"/>	0 dB	Internal
Port 3	Auto	-5.00 dBm	-10.00 dBm	0.00 dBm	<input type="checkbox"/>	0 dB	Internal
Port 4	Auto	-5.00 dBm	-10.00 dBm	0.00 dBm	<input type="checkbox"/>	0 dB	Internal

Power and Attenuation (with True Mode Stimulus) dialog box help

When a True Mode Stimulus is selected in the [Balanced DUT Topology dialog](#), the balanced ports are shown on the top rows of the dialog. The individual PNA port settings are displayed but can not be changed directly.

Port Powers Coupled Check to couple all power settings for Balanced Port 1 and Balanced Port 2. Clear this box to make independent power settings for these logical ports.

State Should be left in Auto for iTMSA measurements. [Learn more about his setting.](#)

Port Power Set power for the balanced port. Power for PNA ports, shown below balanced ports, are set to 3 dB less, or half power. [Power Offset](#) is made in addition to this setting, and on only PNA port 3 and port 4. Offset Power is NOT reflected on this dialog nor on the X-Axis.

Start and Stop Power Available when sweep type is set to [power sweep](#).

Auto Range Check to allow the PNA to select the [optimum attenuation value](#) to achieve the specified test port power. Clear to manually set the attenuation for each port. Type or select the attenuation value in the adjacent Attenuator Control box. When the attenuator setting of a logical port is changed, then the attenuators of the PNA ports associated with that logical port change to the same value.

Leveling Mode Open Loop leveling is available only on ports 1 and 3. [Learn more.](#)

Note: The range of leveled power (ALC range) for balanced ports is 3 dB higher than it is for each individual PNA port. For example, if a PNA source port with 0 dB attenuation will supply leveled power from -30dBm to +15dBm, then the balanced logical port has a range of -27dBm to +18dBm. [Learn more about Leveled Power.](#)

Calibration with iTMSA

Note: Uncalibrated True Mode Stimulus are **NOT** at all accurate.

Perform or recall a SMART Cal exactly like any other 3 or 4-port cal.

Press **CAL**, then [**Cal Wizard**].

- All Cals are performed as single-ended.
- Supports all Fixturing and Port Extension features.
- Supports [Guided Power Cal](#).
- Supports [Enhanced Response Cal](#) and [Source Power Cal](#).

Source Power Cal

Perform a Source Power Cal on:

- **Port 3** for BAL Port 1
- **Port 4** for BAL Port 2

Port 1 and Port 2 power levels are adjusted to these ports in the [initial sweep](#).

Receiver Measurements

Any pair of receivers can be viewed as a ratio using the following dialog.

To select these measurements, press **MEAS**, then [**Receivers**]

S-Parameter	Balanced	Receivers	
	Numerator	Denominator	Source Port
Activate: <input checked="" type="checkbox"/>	<input type="text" value="R3"/>	<input type="text" value="R1"/>	<input type="text" value="Port 1"/>

Create / Change Receiver Measurements (with True Mode Stimulus) dialog box help

Click **Activate**

- For RATIOED measurements, Select a receiver for the **Numerator** and select a receiver for the **Denominator**.
- For UNRATIOED measurements, Select a test port receiver (A, B, C, or D) for the **Numerator**. Reference receiver measurements are NOT accurate. Select **1** for the **Denominator**.

For example, with a Bal-Bal topology, the above selections show a R3/R1 measurement. R (reference) receivers measure the stimulus to the DUT. An R3/ R1 trace shows the difference between the two sources that comprise logical port 1. With Log format, a power offset between the two sources is visible. With Phase format, a phase offset is visible.

Source Port Specifies whether a Differential stimulus or Common mode stimulus is used for the measurement.

- With Source Port **1** or **2** selected, then **Differential** stimulus is used for the unratioed measurement which causes **180° offset** between the sources.
 - Source Port 1 = Differential stimulus on BAL1 port
 - Source Port 2 = Differential stimulus on BAL2 port
- With Source Port **3** or **4** selected, then **Common** mode stimulus is used for the unratioed measurement which causes **0° offset** between the sources.
 - Source Port 3 = Common-mode stimulus on BAL1 port
 - Source Port 4 = Common-mode stimulus on BAL2 port

Differential Input and Output Power Measurement

Equation Editor is used to calculate and display the Differential Input and Output power levels.

This procedure assumes a Bal-Bal topology which requires the following port assignments:

- Balanced Port 1 = PNA ports 1 and 3
- Balanced Port 2 = PNA ports 2 and 4

Procedure

1. **Preset** the PNA

2. Perform a **4-port Guided Power Calibration** if desired.
3. Then press **Trace**, then **New Trace**, then **Receivers**
4. Complete the dialog as follows, then click OK.

S-Parameter	Balanced	Receivers	
	Numerator	Denominator	Source Port
Activate: <input type="checkbox"/>	A /	1.0	Port 1
Activate: <input checked="" type="checkbox"/>	B /	1.0	Port 1
Activate: <input checked="" type="checkbox"/>	C /	1.0	Port 1
Activate: <input checked="" type="checkbox"/>	D /	1.0	Port 1
Activate: <input checked="" type="checkbox"/>	R1 /	1.0	Port 1
Activate: <input type="checkbox"/>	R2 /	1.0	Port 2
Activate: <input checked="" type="checkbox"/>	R3 /	1.0	Port 1
Activate: <input type="checkbox"/>	R4 /	1.0	Port 4

5. The above FIVE traces and the default (S11) trace should be visible on the PNA screen.
6. Click **Tr1** (S11). This is the 'dummy' trace that will become an equation trace.
7. Then press **Analyze**, then **Equation Editor**.
8. In the **Equation** field, enter **(Tr2-Tr4) / sqrt(2)**
9. Press **OK** to close Equation Editor. The **Tr1** trace is now showing Differential Output power.
10. For Differential Input power, activate **Tr3** (C receiver).
11. In Equation Editor, enter **(Tr5-Tr6) / sqrt(2)**. Press **OK**.
12. The **Tr3** trace is now showing Differential Output power.

```
Tr 1 Eq=(Tr2/-Tr4)... , LogM 10.00dB/ 0.00dB   Tr 2 B,1 LogM 10.00dBm/ 0.00dBm
Tr 3 Eq=(Tr5-Tr6)/sqr..., LogM 10.00dBm/ 0.00dBm Tr 4 D,1 LogM 10.00dBm/ 0.00dBm
Tr 5 R1,1 LogM 10.00dBm/ 0.00dBm             Tr 6 R3,1 LogM 10.00dBm/ 0.00dBm
```

The traces used to calculate and display Differential Input and Output Power

Last Modified:

24-Oct-2013	Removed receiver cal
	Added Diff IN/Out Power
5-Aug-2013	Edited front-panel loop note
4-Apr-2013	Removed Guide Power Cal limitation
9-Jul-2012	Added limited mmWave
18-Oct-2011	Added Guided and FP Jumpers to limitations
	Added Restart note
13-Nov-2008	Fixed allowed SE-BAL topology
19-May-2008	MX New topic

Noise Figure Application (Opt 028 and Opt 029)

The Noise Figure Application makes fast, easy, and accurate noise figure measurements.

The information presented in this topic pertains to Noise Figure measurements on BOTH Amplifiers and Converters unless stated otherwise.

- [Noise Figure Options Explained \(029, 028, H29\)](#)
- [Features, Requirements, and Limitations](#)
- [Noise Concepts](#)
- [How the Noise Figure Application Works](#)
- [Scalar Noise Figure Measurements](#)
- [PNA-X Option 029 - Block Diagram](#)
- [The Noise Tuner Switch](#)
- [Noise Parameters that are Offered](#)
- [Using Noise Figure App](#)
 - [Connect Tuner and Noise Source](#)
 - [Create a Noise Figure Measurement](#)
 - [Make Noise Figure Settings](#)
 - [Perform Calibration](#) (separate topic)
 - [Save Noise Data](#)
- [Noise Figure Measurement Tips](#)
- [Using Noise Figure Traces in Equation Editor](#)
- [Noise Model and the Noise Correlation Matrix](#)

See Also

[Noise Figure Calibration](#)

[Noise Figure on Converters \(NFX\)](#)

[Programming commands](#)

[PNA-X Noise Figure Options \(PNA Configuration Guide\)](#)

[High-Accuracy Noise Figure Measurements Using the PNA-X](#)

[Noise Figure and TRL Cal](#)

[See other PNA Applications](#)

Noise Figure Options Explained (029, 028, H29)

See Also: [PNA-X Noise Figure Options \(PNA Configuration Guide\)](#) - Internet connection required

- **029** - (PNA-X ONLY) Includes low-noise receivers and noise tuner bypass switch to enable noise figure measurements to 50 GHz. Also includes Opt 028 capability.
- **028** - Uses standard PNA receivers to measure noise figure. A noise source is NOT used during calibration. Any two ports can be used. Use with DUTs that have sufficiently high gain and noise figure. Additional filtering may be necessary. [Learn more.](#)
- **H29** - Noise Figure on N5244A (43.5 GHz PNA-X model) and N5245A (50 GHz PNA-X model). Includes both:
 - Opt 029 (noise measurements to 26.5 GHz)
 - Opt 028 (noise measurements using standard receiver to 50 GHz) . [See H29 specifications.](#)

50 GHz Noise Figure Receivers

Beginning in October 2012, Option 029 (low-noise receivers) is available in the N5244A (43 GHz), N5245A (50 GHz), and N5247A (67 GHz) models.

- These models (with option 029) include a built-in (internal) Noise Tuner which can be selected at the beginning of the [Noise Figure Calibration](#). The Noise Tuner switch is managed differently than in 26.5 GHz models. [Learn more.](#)
- When used with the N5247A, Noise Figure measurements between 50 GHz and 67 GHz are possible using the standard receivers (Opt 028) and a 67 GHz Power Sensor. [See Opt 028 Measurement tips.](#)
- [See limitations of the 50 GHz Noise Source.](#)

Noise Figure Application Features

- Cold noise method includes correction for imperfect system source match for highly accurate noise figure measurements.
- With Opt 029 you can measure devices with noise figure values ranging from about 0 to 50 dB and devices

with GAIN ranging from about -40 to +60 dB. [Learn more.](#)

- With Opt 028 you can also measure noise figure using standard PNA receivers to 67 GHz. [Learn more.](#)
- Measure noise figure of frequency translating devices. [Learn more.](#)
- During calibration, ENR values are interpolated for frequencies between the supplied data points.

Noise Figure Application Requirements

- Noise Tuner - Required for vector noise figure measurements. Not required for [scalar noise figure](#).
 - ONLY the N4690 Series ECal modules are supported. The N4691B m-f is recommended.
 - Opt 029 provides an additional cable and adapter to connect the ECal module to the front-panel connectors. [Learn more.](#)
 - A built-in Noise Tuner is provided with the 50 GHz noise receiver models. [Learn more.](#)
- Power Meter - Required when calibrating [NFX](#) (Noise Figure on Converters).
- Recommended: An accurate thermometer. [Learn more.](#)

Noise Source

A Noise Source is NOT required to calibrate the Opt 029 Noise Receivers. Instead, the Noise Receivers can be calibrated using a calibrated PNA source. [Learn more.](#)

When using a Noise Source, the following requirements apply:

- The 346C Noise Source (recommended) produces ENR values to 26.5 GHz.
- The 346B Noise Source can be used up to 18 GHz.
- The 346A Noise Source can also be used up to 18 GHz, but requires more [averaging](#) for calibration.
- The 346C K01 (50 GHz) Noise Source typically has about 6 dB of ENR at 50 GHz which may NOT yield an adequate calibration, depending on how many noise averages are used. An alternative approach calibrates the noise receivers using a power sensor-based method. Select **Use Power Meter** for the noise figure calibration. [Learn more.](#)
- An adapter may be necessary to connect the Noise Source to the PNA port 2 reference plane during [calibration](#). Cal Kit (or second ECal module) with same connector type and gender as DUT connectors.

Limitations with the Noise Figure Application

The following features are NOT supported in a noise figure channel:

- [FCA \(opt 083\)](#) or [Frequency Offset \(opt 080\)](#).
- [Analog sweep](#). All frequency sweeps are STEPPED.
- Independent IFBW, Power Levels, or Sweep Time in a [segment table](#) is NOT supported.
- [External Test Set Control](#) (Opt 550 or 551)
- [Receiver calibration](#).
- [Enhanced Response Cal](#)
- [ECal User Characterization](#).
- Some [Fixturing Features](#)
- [Auto Port Extensions](#)
- [Auto Formatted Citifile](#) data.
- [External DC Devices](#)
- [Pulsed](#) noise figure measurements are supported with the following limitations:
 - Minimum 300 microsecond pulse width using 24 MHz noise bandwidth
 - Narrower noise bandwidths cause larger minimum pulse widths
 - A drop-out may occur at start of sweep and at 3 GHz. This is corrected by a 1 ms pulse width at 24 MHz Noise BW.

Noise Concepts

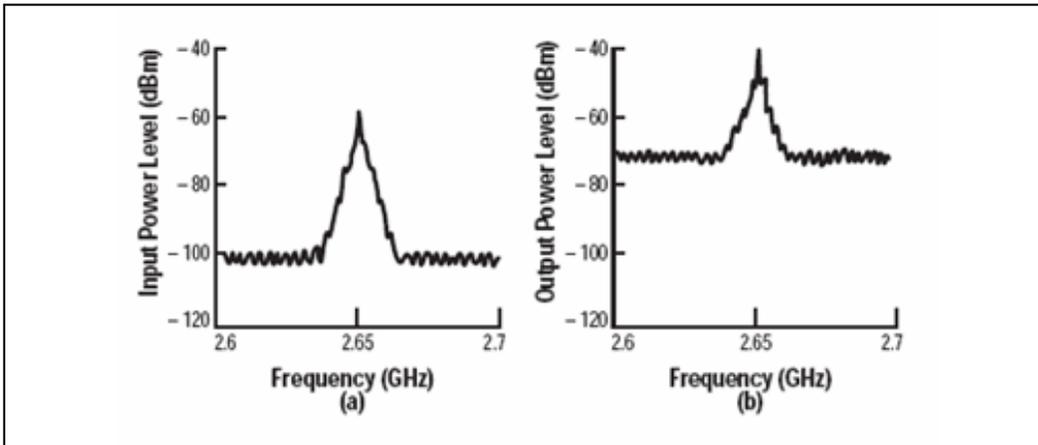
The following conceptual information is a short summary taken from the [Agilent Noise Figure App Note 57-1](#).

All electronic circuits have some degree of random noise. The most common form is thermal noise, which increases as the temperature of the circuit increases.

The signal-to-noise (S/N) ratio of components in a communications system is a very important parameter. To improve the S/N ratio, it is usually easier and more cost-effective to reduce noise than to increase signal power. In order to reduce noise, an accurate method to measure noise is required.

Noise Figure

Noise Figure is the degradation in the signal-to-noise ratio as a signal passes through a device. For example, in the following images:



(a) At the INPUT of the DUT:

The noise floor is -100 dBm, the signal is at -60 dBm, **40 dB** above the noise floor.

(b) At the OUTPUT of the same DUT:

The gain has boosted the signal AND the noise floor by 20 dB.

The DUT then added 10 dB of its own noise.

The output signal is now only **30 dB** above the noise floor.

Since the degradation in signal-to-noise ratio is 10 dB, the DUT has a **10 dB noise figure**.

For consistency, noise measurements are calculated as if using a 1 Hz bandwidth, although measurements are almost always made at higher bandwidths.

The following formula shows the lowest possible noise power in dBm at 290° K (room temperature). The only way to measure noise lower than this is to make the measurement at a lower temperature.

- $P = 10\text{LOG}(4.0 \times 10^{-21} \text{ watts}/.001 \text{ watt})$
- $P = -174 \text{ dBm} / \text{Hz}$

How the Noise Figure Application Works

The goal of the noise figure application is to accurately measure the noise that is generated by the DUT. This may be done using special low-noise receivers or using the standard PNA receivers depending on whether the PNA has Options 029 or 028. [Learn more.](#)

The standard receivers are always calibrated using a power meter and a measurement of the receivers effective noise bandwidth. The low-noise receivers can be calibrated using either a characterized noise source or using the same process as a standard PNA source. [Learn more about the noise calibration process.](#)

Some noise measurement error is caused by a poor source match presented to the DUT input. Therefore, during every measurement, the noise figure application uses an ECal module to present at least four different impedances at the input of the DUT. This "Noise Tuner" is connected to the PNA port 1 front-panel loops that are in the PNA internal source path (see [block diagram](#) below). From the measurements at various impedance states, the PNA calculates the noise out of the DUT as though the PNA were exactly 50 ohms. No assumptions are made regarding the input impedance of the DUT.

Here is how a vector noise figure measurement is made using Option 029. The sweep numbers are annotated on the PNA display as they occur.

1. With the noise tuner in the THRU state, S-parameter measurements are made to accurately characterize the gain of the DUT. This requires sweeps in both forward and reverse directions. (sweep #1 and #2).
2. The noise measurements are performed next. PNA source power is turned OFF and the noise tuner is switched to the first impedance state.
3. At each frequency, the noise receiver samples a large number of readings in order to attain **one** valid measurement. If [Noise Averaging](#) is selected, the specified number of measurements are made and averaged together to obtain one noise measurement. This continues for all frequencies (sweep #3).
4. The next noise tuner impedance state is switched IN and the noise measurements in step 3 are repeated. This occurs until measurements are made at all impedance states. At least four impedance states must be used. (sweeps #4, #5, #6+)
5. Calibration error terms are applied and calculations made to simulate the measurement with a perfect 50 ohm input impedance. The sweep result is plotted on the PNA display.
6. The PNA begins sweeping again with step 1.

Scalar Noise Figure Measurements

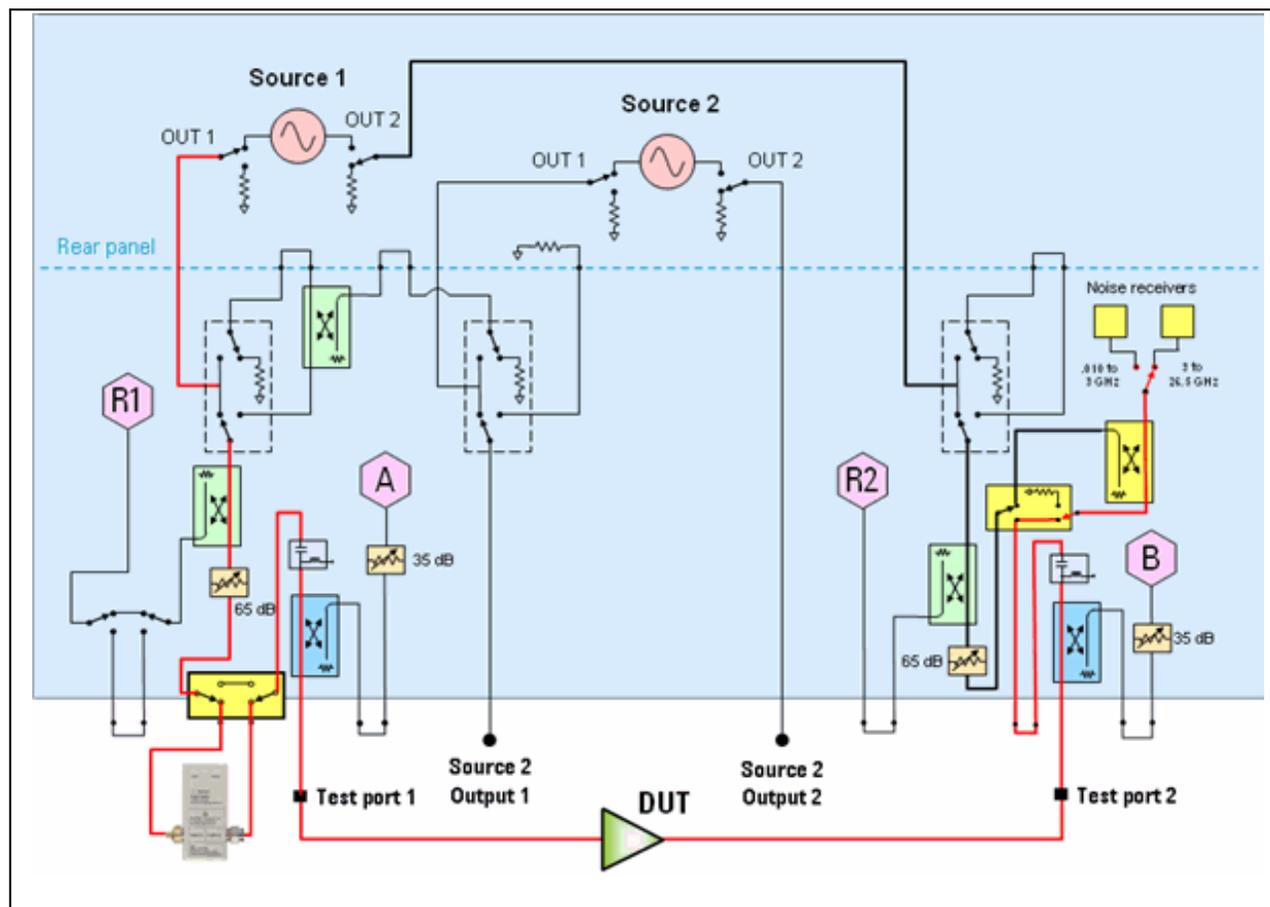
Scalar noise figure measurements can be made beginning with PNA Rev. A.08.50.

As described above, the noise tuner is switched to at least four different impedance states before a sweep is plotted. These sweeps are NOT made in a scalar noise figure measurement, resulting in much faster measurements. Of course, a scalar noise figure measurement is NOT as accurate as a vector noise figure measurement because scalar noise figure measurements assumes that all impedances are 50 ohms. Measurement accuracy can be improved by adding an attenuator as close to the DUT input as possible. This improves the effective system source match. The effect of the attenuator loss is removed during the calibration process.

With scalar noise figure, it is not necessary to connect the noise tuner. If a noise tuner remains connected, it is switched to the THRU state for scalar noise figure measurements. This results in a small amount of loss which slightly degrades measurement accuracy. To increase measurement accuracy, manually switch the noise tuner switch to the INTERNAL position. [Learn how.](#)

Select **Scalar Noise** at the first page of a [Noise Figure calibration](#).

PNA-X Option 029 - Block Diagram with Noise Figure components



26.5 GHz Noise Figure Components are shaded yellow

- At test port 1 front-panel loops, a **noise tuner bypass switch** connects the noise tuner (ECal module) in series with Source1 providing several different input impedances. [Learn more about managing the Noise Tuner switch.](#)
- At test port 2, a **switch** and **coupler** to route RF from the DUT output to **two noise receivers**. The appropriate receiver is automatically switched as required for the frequency being measured.

See Also: [50 GHz Noise Figure - Built-in Tuner](#) switch below.

make switch settings. However, the switch is can be changed in any of the following ways:

- Using the [path configuration](#) dialog.
- Using the following commands:
 - **SCPI** - [SENS:PATH:CONF:ELEM:STAT](#) "Port1NoiseTuner", "Bypass" (or "Internal")
 - **COM** - [PathConfiguration.Element](#)("Port1NoiseTuner").Value = "Bypass" (or "Internal")
- Manage the built-in Noise Tuner impedance states using the SCPI and COM commands and [Element / Value settings](#).

Using the Noise Figure Application

Use the following general procedure to make noise figure measurements:

1. [Connect Tuner and Noise Source](#).
2. [Create a Noise Figure Measurement](#).
3. [Make Noise Figure Settings](#).
4. For Opt 029 and H29, copy your Noise Source ENR file to the PNA C:/Program Files/Agilent/Network Analyzer/Noise folder.
5. [Perform Calibration](#)
6. Connect the DUT. [Learn more about DUT input and output ports](#).
7. Measure Noise Figure.
8. **Optional** Click **File**, then **Save** to save noise figure data. [Learn more](#).

Connect Noise Tuner and Noise Source

1. Connect the **noise source** to the [28V connector](#) on the PNA rear panel. **NOT required for Opt 028**. The Noise Source is turned ON and OFF automatically as needed during a calibration. Connect the noise source to Port 2 reference plane when prompted during calibration.
2. Connect the **noise tuner** (ECal module). [See Noise Tuner requirements](#).
 - a. On the PNA front panel, remove the **Port 1** jumper cable SOURCE OUT / CPLR THRU. Opt 028 allows noise figure measurements using any two PNA ports.
 - b. Connect the noise tuner to the front-panel jumpers for the source (DUT input) port.



See the [PNA Configuration Guide](#) for recommended ECal modules, cables, and adapters.

Create a Noise Figure Measurement

1. On the PNA front panel, press **Meas**, then **[Measurement Class]**
2. Select **Noise Figure Cold Source**, then either:
 - **OK** delete the existing measurement, or
 - **New Channel** to create the measurement in a new channel.
3. A noise figure measurement is displayed. The following shows how to select or change displayed parameters.

Noise Parameters

Several noise parameters, as well as standard parameters, can be measured in the same Noise channel.

How to add Noise Parameters

1. Create a Noise Figure channel.
2. Then do the following:

Using front-panel HARDKEY [softkey] buttons

1. Press **TRACES**
2. then **[New Trace]**
3. then select a parameter

Using a mouse with PNA Menus

1. Click **Trace/Chan**
2. then **New Trace**
3. then select a parameter

How to CHANGE Noise Parameters

1. Create a Noise Figure channel.
2. Select the parameter to change.
3. Then do the following:

1. Press **MEAS**

2. then select a parameter

1. Click **Response**

2. then select a parameter

Programming Commands

Noise Measurements that are offered

The following three categories of noise measurements can be made with the PNA:

1. **Noise Figure** is the amount of noise that the DUT is adding in a 50 ohm test setup. This is explained in detail in [Noise Concepts](#).
2. [Noise Power Parameters](#) show the amount of noise coming out of the DUT in a 50 ohm test setup. With gain measurements of the DUT, these noise power parameters are used to calculate noise figure.
3. [Noise Parameters](#) are models of the noise that is generated in a DUT, similar to how S-parameters model how RF flows through a DUT.

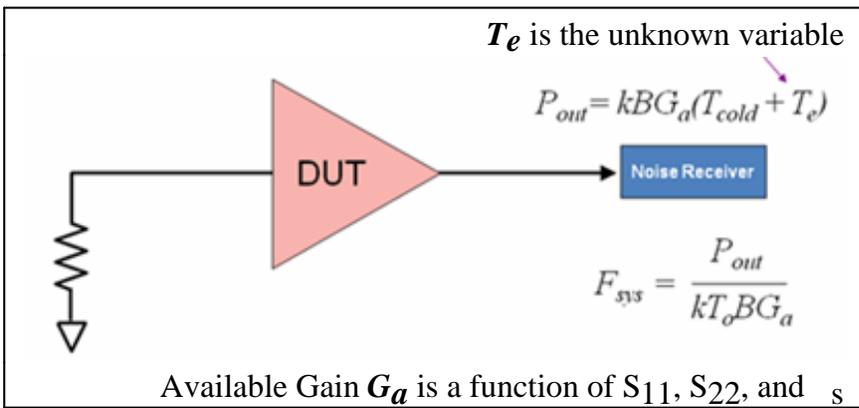
-
- **Noise Figure (NF)** - Explained in [Noise concepts](#).
 - **Excess Noise Ratio** - Select when measuring the noise source. Compare with the ENR table to validate accuracy of the system. ENR is calculated as:

$$\text{ENR (in dB)} = 10 \log_{10}((T_{\text{hot}} T_{\text{cold}}) / T_0), \text{ where } T_0 = 290\text{K.}$$

Learn more about the ENR table and Noise Source. [Learn more about Noise Source ENR measurements.](#)

- **T-Effective** - The effective temperature, in Kelvin, of the measured noise level. For example:

$$290^\circ \text{ K} = -174 \text{ dBm/Hz.}$$



Noise Power Parameters

The Noise Power parameters below are offered in the following two formats:

- **Available Noise Power** The calculated power that is based on an ideal impedance match at the output of the DUT. These parameters have always been offered in the PNA noise figure App.
 - **Incident Noise Power** - An 'I' is appended to the end of the Available Noise Power parameter. The calculated power into a perfect 50 ohm noise receiver, regardless of the output impedance of the DUT.
-
- **SYSNP / SYSNPDI** - System Noise Power Density: Total noise power available at the ADC, including the noise contributed by both the DUT and the internal noise receiver. This is generally expressed as an absolute power measurement in dBm, but can also be expressed in Watts or Kelvin.

$$\text{dBm} = 10 \log_{10}(k * T * B * 1000)$$

where:

k = Boltzmann's constant

T = the measured noise temperature

B = bandwidth

1000 = conversion from milliwatts

- **SYSRNP / SYSRNPI** - System Relative Noise Power: The noise temperature of the combined DUT and receiver relative to 290 Kelvin. This is generally reported as a ratio in dB. Therefore a perfectly quiet device would render a trace at 0 dB.

$$\text{dB} = 10 \log_{10}(T/290)$$

- **DUTNP / DUTNPDI** - DUT Noise Power Density: When correction is ON, this trace exhibits the available noise power, best described as the maximum power available from the DUT where the impedance of the noise port is equal to the output match of the DUT. To be more precise, this occurs when the noise port

match is equal to the conjugate of the output match of the DUT. The noise power contributed by the receiver is removed.

When correction is OFF, the trace exhibits what is more accurately described as delivered power. Delivered power is the power actually seen by the ADC. Any mismatch between the receiver and the DUT is ignored. The noise power contributed by the receiver is removed.

This measurement is generally expressed in dBm, normalized to a 1 Hz bandwidth. For convenience, marker and trace readout shows **dBm**.

You could display the power in a different bandwidth using [Equation Editor](#).

$$\text{dBm/Hz} = 10 \log_{10} \left(\frac{\text{DUT Temperature} - \text{Receiver Temperature}}{1000} \right) * B *$$

where:

B = bandwidth

1000 = conversion from milliwatts

- **DUTRNP / DUTRNPI** - DUT Relative Noise Power: This measurement is rendered as a ratio of the DUT temperature to 290 Kelvin. It is generally expressed in dB. The same comments apply with respect to available versus delivered power as described above for DUTNPD.

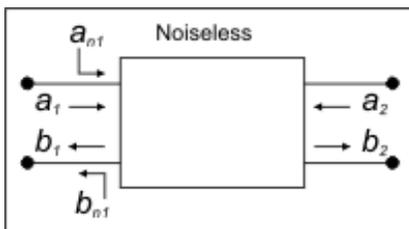
$$\text{dB} = 10 \log_{10} (\text{DUT Temperature} - \text{Receiver Temperature})$$

Noise Model, Noise Parameters, and the Noise Correlation Matrix

Noise Parameters are models of the noise that is generated in a DUT, similar to how S-parameters model how RF flows through a DUT.

Noise Model

The noise wave model of any linear 2-port network may be represented by the following image:



This shows a noiseless 2-port network with noise waves (a_{n1} and b_{n1}) added to the input terminals. The a_1 a_2 and b_1 b_2 are standard S-parameter waves.

The noise correlation matrix relates to the noise waves as follows:

$$C_i = \begin{bmatrix} \overline{|a_{n1}|^2} & \overline{a_{n1} b_{n1}^*} \\ \overline{b_{n1} a_{n1}^*} & \overline{|b_{n1}|^2} \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$$

Where:

- $\overline{|a_{n1}|^2}$ and $\overline{|b_{n1}|^2}$ are time-averaged noise power in 1 Hz bandwidth.
- $\overline{a_{n1} b_{n1}^*}$ and $\overline{b_{n1} a_{n1}^*}$ are time-averaged cross correlation terms, correlation of a_{n1} to b_{n1} .
- Overbars represent time-averaging
- Star superscripts represent complex conjugation

Noise Parameters

- **GammaOpt** (Optimum Complex Reflection Coefficient) - The optimal impedance for the noise figure measurement. Select the [data format](#) to display Gamma-Opt-Magnitude and Gamma-Opt-Phase (most common), or Gamma-Opt-Real and Gamma-Opt-Imaginary.
- **NFmin** - The minimum noise figure that occurs at GammaOpt.
- **Rn** (Noise Resistance) - Specifies the rate of change of the level of noise when varying the source impedance.
- **NCorr_11, NCorr_21, NCorr_12, NCorr_22** - The NCorr_11 and NCorr_22 terms are effective noise temperature, normalized to 290 K. Both terms are time-averaged, noise-wave powers referred to the input of the DUT, where NCorr_11 is the forward wave (noise going through the device towards the output), and NCorr_22 is the reverse noise wave (noise coming out of port 1 of the DUT, going back towards the source).
 - To convert to available noise power, multiply the terms by $290 \cdot k \cdot B$ where:
 - k = Boltzmanns constant
 - B = system bandwidth

Standard Parameters that are offered (Amplifiers-only)

- **S-parameters:** S11, S21, S22, S12
- **Unratioed parameters** using the following notation: (Receiver, source port). These parameters REPLACE the active measurement. To do this (from front-panel ONLY), press **MEAS**, then **[More]**, then **[Receivers]**.
 - (R1,1), (R2,2), (A,1), (A,2), (B,1), (B,2)

Save Noise Data

To save noise data, click **File**, then **Save Data As** Then select from the following **Save As Types**:

- **(*.prn), (*.cti), (*.csv), (*.mdf)** - Noise Figure data can be saved ONLY with these choices. PRN saves only the active trace. CITI formatted, CSV Formatted, and MDF can save all displayed traces. [Learn more about these formats.](#)
- **(*.s2p)** - Saves S-parameter data only after performing a Noise calibration. This data is saved regardless of which noise measurement is active or displayed. [Learn more about *.s2p data.](#)
- **Trace and Noise parameter (*.s2p)** - Saves S-parameter data, then the [Noise Parameters](#). This data is saved regardless of which noise measurement is active or displayed. When the vector calibration is not enabled or if the noise parameters are not realizable, then the noise parameters have no calculated value. In this instance, the following values are displayed instead:
 - **GammaOpt** = 0
 - **NFmin** = raw noise figure
 - **R_n** = $Z_0 / 4 * (F - 1)$. This equation is how R_n is currently calculated for ill-conditioned data. F is the noise factor where F is related to the noise correlation value ct₁₁ and the normalized noise temperature T_n by $F = 1 + ct_{11} = 1 + T_n$ so that $R_n = (Z_0 / 4) * ct_{11}$
- **NoiseCorr (*.nco)** - Saves Noise Correlation data regardless of which noise measurement is active or displayed. The *.nco file is a noise correlation matrix expressed in T-parameter form (**Ct11, Ct21, Ct12, Ct22**). These parameters are exactly the same as the [Noise parameters](#) **NCorr_11, NCorr_21, NCorr_12, NCorr_22** that can be displayed as traces.
- When the vector calibration is not enabled, this data is set to -200 dBm.

How to start the Noise Figure Setup dialog

To provide quicker access, use the Setup softkey. [Learn how.](#)

Using front-panel HARDKEY [softkey] buttons

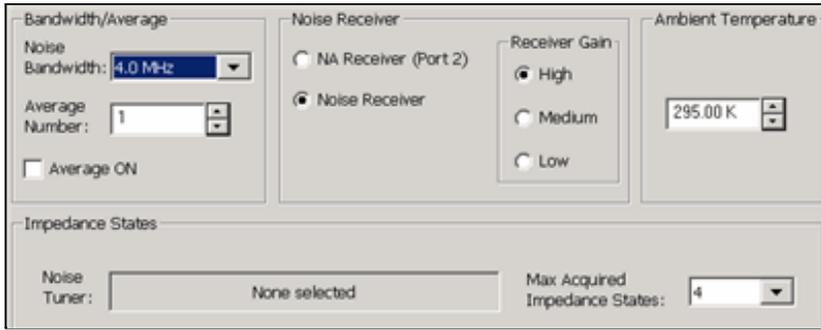
1. Press **FREQ**
2. then **[Noise Figure Setup]**

Using a mouse with PNA Menus

1. Click **Response**
2. then **Measure**
3. then **Noise Setup**

[Programming Commands](#)

Noise Figure Setup dialog box help



Note: In this topic, the term **Jitter** is used to describe the trace-to-trace fluctuations in a measurement. In other topics, this is called 'trace noise'.

Bandwidth/Average

The following two settings work together to achieve the optimum balance of measurement accuracy versus speed:

Noise Bandwidth Increase the bandwidth to reduce the amount of trace noise on the noise power or noise figure measurement (jitter). However, a wider setting reduces the frequency resolution of the measurement. The noise bandwidth setting should always be smaller than the bandwidth of the DUT. The noise bandwidth setting is used only while measuring noise powers, and is independent from the [IF bandwidth setting](#) used to measure S-parameters. Noise figure is calculated from noise power and S-parameter measurements.

The calibration and measurement should be performed using the SAME noise bandwidth. When the noise bandwidth is changed after calibration, noise figure measurements can change by 0.5 dB or more, depending on the DUT frequency range, gain, and noise figure.

Note: The Noise Bandwidth may be adjusted automatically at low frequencies according to the following table. At each data point frequency, if the specified Noise BW is higher than that shown in the table, the Noise BW is set to the max value in the table.

	RF Bands			
	10 to 25 MHz	25 to 60 MHz	60 to 150 MHz	Above 150 MHz
Noise Bandwidths Allowed	.8 MHz	.8 MHz	.8 MHz	.8 MHz
	2 MHz	2 MHz	2 MHz	2 MHz
		4 MHz	4 MHz	4 MHz
			8 MHz	8 MHz
				24 MHz

Note: [Use Power Meter calibration method](#) is **NOT** available when the Noise Bandwidth is **8 MHz** or **24 MHz**.

Average Number Increase the number of averages to reduce jitter. This also increases measurement speed. For maximum accuracy, use the following recommendations for the noise calibration. When using the noise receivers, 10 noise averages is recommended. When using the standard receivers, at least 100 averages are recommended.

During a measurement, the gain of the DUT helps overcome the noise of the PNA receivers, so the number of noise averages can be reduced to improve measurement speed with minimal or no degradation to measurement accuracy.

Noise Receiver

NA (Network Analyzer) Receiver (Opt 028) - Use a standard PNA receiver to measure noise figure.

- Connect the DUT to any PNA ports. For vector noise figure measurements, [connect the noise tuner](#) to the source port.
- The gain plus noise figure of the DUT minus cable loss must be at least 40 dB (**G+ NF - Loss > 40 dB**). This ensures that there is sufficient DUT noise power for the PNA to measure. [Learn more.](#)
- Additional filtering may be required. [Learn more.](#)

Noise Receiver (Opt 029) - Use internal low-noise receivers to measure noise figure.

- Opt 029 measures devices with noise figure values ranging from about 0 to 50 dB and devices with GAIN ranging from about -40 to +60 dB. [Learn more.](#)
- Amplifiers with higher gain can be measured by adding an attenuator to the output of DUT and using [fixture de-embedding](#) to remove the attenuator loss. An alternative for measuring high-gain devices is to use the standard receivers (Opt 028) as they have a higher compression level.

Receiver Gain

This setting is NOT available when Noise Receiver is set to **NA (Network Analyzer) Receiver (Opt 028)**.

With knowledge of your DUT gain, set the appropriate amount of receiver gain in order to optimize the power level at the noise receiver.

The following values reflect the SUM of the DUT gain (dB) **PLUS** NF (dB). For example: DUT gain = 20 dB; NF = 10 dB; SUM = 30 dB.

- Select **High** if the SUM is relatively low (<30 dB).

- Select **Medium** if the SUM is about average (20 dB to 45 dB).
- Select **Low** if the SUM is relatively high (>35 dB).

There is considerable overlap in these settings. Because all three gain settings are calibrated with each Noise Calibration, this setting can be changed after calibration to achieve the least amount of jitter without overpowering the noise receiver.

One of following messages appears when too much power is detected at the noise receiver:

- **Compression in noise receiver: excess signal** - The noise receiver is likely compressing. NF results are possibly not accurate. Select a lower gain setting.
- **Compression in noise receiver: gain has been limited** - The gain has been limited to avoid damage to the receiver. NF results are NOT accurate. Select a lower gain setting.
- **ADC over-range in noise receiver: excess signal** - Often caused by a CW signal, an oscillation, or LO feedthru during an NF measurement. Find and correct the cause, or try a lower gain setting.

Only ONE gain setting can be used for the entire frequency range of your noise measurement.

Therefore, it may be necessary to use two noise channels with different frequency ranges and gain settings to achieve the very highest noise figure accuracy.

Ambient Temperature

Enter the room temperature at the time of the measurement, in Kelvin. For best results, use a thermometer to read the temperature at the PNA test port 1 or the DUT input cable.

This ambient temperature number has an inverse relationship to the noise figure. When using the effective noise temperature (Te) format, a 3 degree increase in the ambient temperature will make the overall measurement result drop 3 degrees.

Impedance States

Noise Tuner Displays the ECal module to be used as a noise tuner. Select the Noise Tuner during calibration on the [Select Cal Method](#) dialog.

Max Acquired Impedance States Select the number of impedance states in which to make noise measurements. At least FOUR impedance states are required. [Learn more](#)

Frequency Tab - Noise Figure dialog box help

The screenshot shows the 'Frequency' tab of the 'Noise Figure' dialog box. It features a 'Sweep Type' section with radio buttons for 'Linear Sweep' (selected), 'Log Sweep', 'Power Sweep', 'CW Frequency', and 'Segment Sweep'. Below this is the 'Sweep Settings' section with several input fields: 'Number Of Points' (201), 'IF Bandwidth' (1.000 kHz), 'Start' (10.000000 MHz), 'Stop' (26.50000000 GHz), 'Center' (13.25500000 GHz), and 'Span' (26.49000000 GHz). Each field has a small up/down arrow icon.

These settings can also be made from the normal PNA setting locations. Click links below to learn how.

Sweep Type

Choose a sweep type. [Learn more.](#)

Segment Sweep Notes:

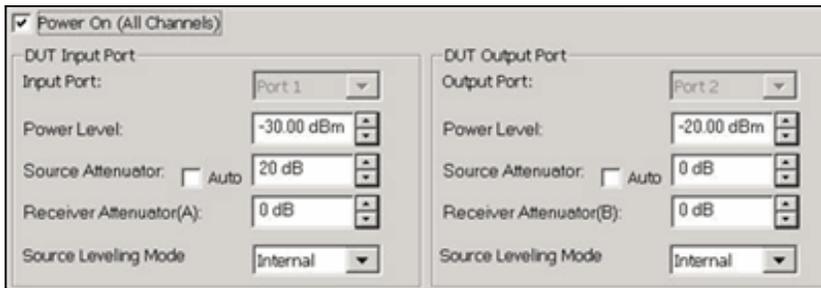
- The segment table shown on the dialog is **'READ-ONLY'**.
- Learn how to [Create and edit the Segment Sweep table.](#)
- **Independent IFBW** and **Power** are NOT available.
- [X-axis point spacing](#) is available beginning with A.09.10.

Sweep Settings

Click each to learn more about these settings.

- [Number of points](#)
- [IF Bandwidth](#) This setting is important for improving noise measurement accuracy. [Learn more.](#)
- [Start / Stop, Center / Span](#) frequencies.

Power Tab - Noise Figure dialog box help



Note: S-parameter power settings are critical for accurate noise figure measurements. [See Noise Figure Measurement Tips.](#)

Configures RF power settings for the S-parameter measurements that occur before noise measurements. Input power to the DUT is turned OFF during noise measurements.

These settings can also be made from the normal [Power setting](#) locations.

Power ON (All channels) Check to turn RF Power ON for all channels.

DUT Input Port

Opt 028 - Select a PNA port to be connected to the DUT input.

Opt 029 [Scalar Noise Figure](#) - Select a PNA port other than port 2.

Opt 029 Vector Noise Figure - The DUT input CAN be connected to any PNA port other than port 2. However, without a noise tuner bypass switch, measurements on other channels that use the same source port will always go through the noise tuner. The noise tuner must be connected to the source loop of the selected port.

Note: Input power levels are critical for accurate noise figure measurements. [Learn more.](#)

Power Level The input power to the DUT during S-parameter measurements.

Source Attenuator Auto Check to automatically select the correct attenuation to achieve the specified input power. Clear, then select attenuator setting that is used achieve the specified Power Level. [Learn more about Source Attenuation.](#)

All PNA channels in continuous sweep must have the same attenuation value. [Learn more.](#)

Receiver Attenuator Specifies the receiver attenuator setting for port 1.

Source Leveling Specifies the leveling mode. Choose Internal. Open Loop should only be used when doing [Wideband Pulse measurements](#) (not available with Noise figure measurements).

DUT Output Port

Opt 028 - Select a PNA port to be connected to the DUT output.

Opt 029 - Connect the DUT output to PNA port 2.

Output Power Sets power level in to the output port for reverse sweeps. Port power is automatically uncoupled. Reverse sweeps are always applied to the DUT when Full 2-port correction is applied. [Enhanced Response Cal](#) is NOT available for noise figure measurements.

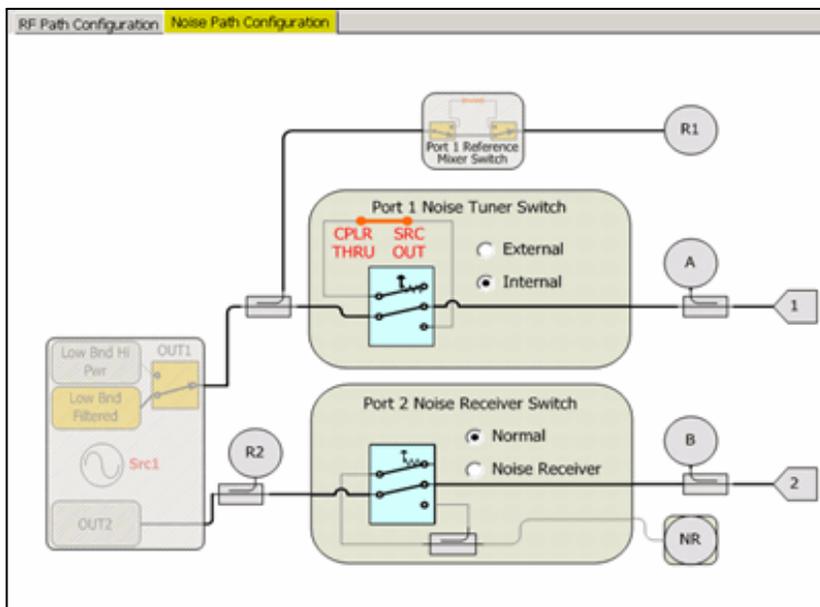
Source Attenuator Specifies the source attenuator setting for reverse power.

Receiver Attenuator Specifies the receiver attenuator setting for the output port. [Learn more about Receiver Attenuation.](#)

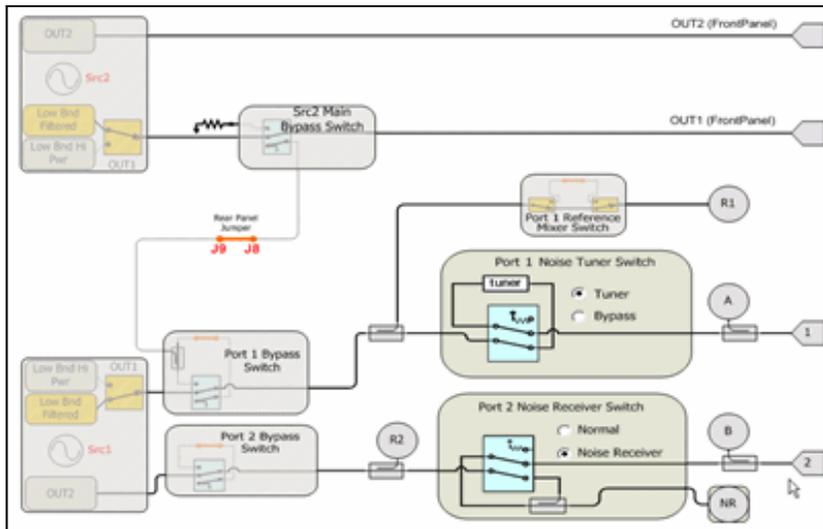
Source Leveling Specifies the leveling mode. Choose Internal.

Noise Path Configurator - dialog box help

26.5 GHz Models



50 GHz Models



Port 1 Noise Tuner Switch (Opt 029)

26.5 GHz

The orange line between CPLR THRU and SRC OUT represents the Noise Tuner.

- **External** selects the external Noise Tuner for making noise figure measurements.
- **Internal** bypasses the external Noise Tuner

See [Important Notes](#) about managing this switch.

50 GHz Models

- Tuner - Represents the built-in Noise tuner.
- Bypass - Bypasses the built-in tuner

Port 2 Noise Receiver Switch (Opt 029 All models) allows you to make Noise Receiver measurements.

To prevent premature wear on the above two Noise switches, the PNA does not allow these switches to be thrown when sweeping a Noise channel and non-Noise channel. To make noise figure measurements and non-noise figure measurements in different channels and continuously trigger both, set these switches to the same state as the Noise channel:

- With the **non-noise figure channel** active, go to [Noise Path Configurator](#).
- Set Noise Tuner switch to **External**. This routes source power to the front-panel loops, and to the Noise Tuner when connected. Use [CONT:ECAL:MOD:PATH:STATE](#) to set the internal state of the Noise Tuner to THRU, which creates a small amount of additional loss in the source path.

- Set Noise Receiver Switch to **Noise Receiver**.

Noise Figure Measurement Tips

Note: In this topic, the term **Jitter** is used to describe the trace-to-trace fluctuations in a measurement. In other topics, this is called 'trace noise'.

Option 029

[See Opt 028 \(NF with Standard Receiver\)](#)

- Measures devices with noise figure values ranging from about 0 to 50 dB and devices with GAIN ranging from about -40 to +60 dB.
- Highest noise figure accuracy is attained when the sum of device noise figure + GAIN is between 0 dB to +70 dB.
- For highest noise figure accuracy and stability, there should be the least amount of electrical loss possible between the DUT output and PNA Port 2.

Power level at the DUT Output - Opt 029

S-parameters are used to measure the gain of the DUT before each series of noise measurements. Jitter in the S-parameter measurements corresponds directly to jitter in the noise measurements.

For best noise figure accuracy, the power level out of the DUT should be between 15 dB and 20 dB below the compression point of the DUT during the S-parameter portion of the noise figure measurement.

To reduce jitter, the power level at the B receiver (port 2) should be above approximately -20 dBm. Much below this level, S-parameter measurements have more jitter. Power must be below +10 dBm as the B receiver starts to compress at this point, although there is no warning or annotation that shows this condition is occurring in S-parameter measurements.

The best way to monitor power at the B receiver is to display a [B,1 measurement](#). With your DUT in place and powered ON, change the input power to the device and note the power at the B receiver.

- For low-gain DUTs, use 5 dB of source attenuation to improve the uncorrected match of port 1.
- For high-gain DUTs, source and receiver attenuation may be required. Use the lowest possible attenuation values.

Attaining the optimum power level **during calibration** can also be challenging since calibration is performed without the DUT in place. Because of this, it is often necessary to set source power higher during the calibration than during the measurement. This will cause the ['CA' annotation](#) on the status

bar. Measurement results are accurate as long as the step attenuators and other configuration switches are in the same position and all receivers remain in their linear range (below +10 dBm).

It is best to find the optimum power and attenuation settings for both the calibration and subsequent noise measurements **before** performing a calibration.

IF Bandwidth

Jitter is further reduced by narrowing the IF bandwidth. If the calibration needs to be performed at a low source power, or with receiver attenuation due to high DUT gain, the IF bandwidth should be reduced during the calibration to reduce jitter. The IF bandwidth can then be increased to improve measurement speed. The [CA annotation](#) can be ignored when changing IFBW after calibration.

Noise Settings

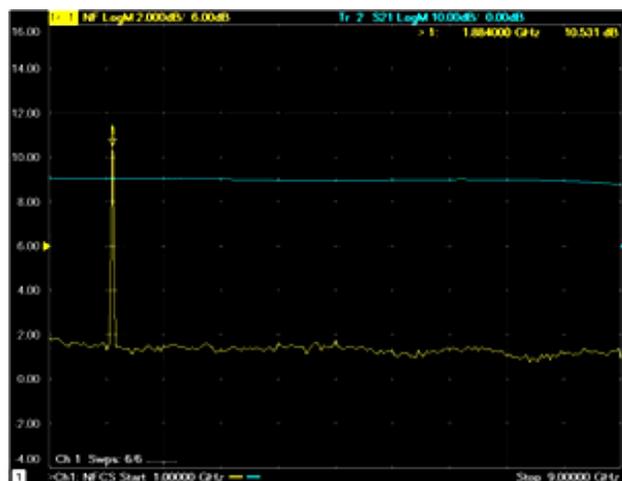
See [Noise Figure dialog box help](#) for a complete description of these important settings.

Temperature

Noise Figure measurements are extremely sensitive to temperature. As such, there are two settings that require an accurate temperature measurement: At the [DUT input](#), and at the [Noise Source connector](#).

Interference

When measuring the noise figure of an unshielded device, like an amplifier on a printed-circuit board, it is very common to pick up interference from external signals such as cellular phones, wireless LAN, or mobile radios. This interference shows up as non-repeatable spikes in the measurement, as shown below.



Usually, the interference adversely affects the noise figure measurement only at the frequency where it occurs. However, if the interference is large enough and present all of the time, it can cause the noise receivers to compress, which results in inaccurate measurements at many frequencies. In this case, the

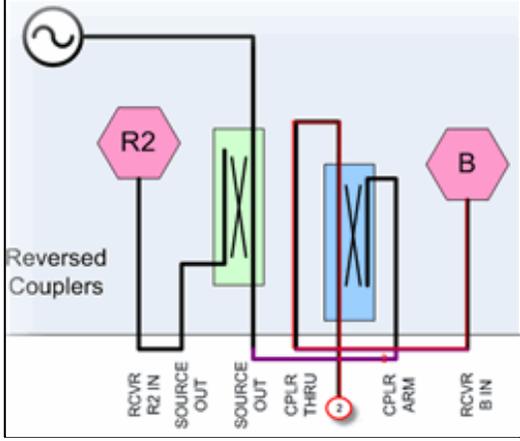
noise figure measurements should be done in a shielded environment like a screen room.

Option 028

Noise Figure of PNA receiver - Option 028 gives you the flexibility to measure noise figure using a standard PNA receiver. For best measurement accuracy, the DUT excess noise power, which is gain plus noise figure minus cable loss in dB ($G + NF - \text{Loss}$), should meet or exceed the noise figure of the receiver. This is generally not a problem with very high-gain devices such as converters with approximately 60 dB of gain.

If your DUT is NOT a very high-gain device, you can re-configure the PNA front panel loops to increase receiver sensitivity.

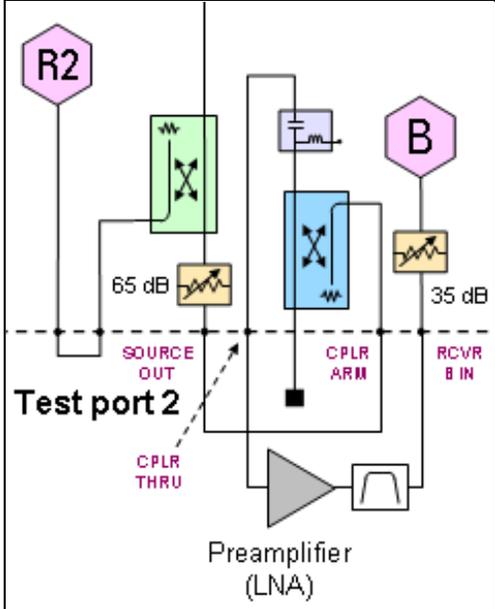
Re-configuring the front panel loops - This configuration reverses the main arm and coupled arm of the test-port coupler (see following images). This increases the signal to the receiver port by about 15 dB, while lowering the available port power by the same amount. This is a good tradeoff for noise figure measurements.

 <p>Block diagram showing port 2 thru coupler main arm to B receiver.</p>	 <p>Configure the receiver port front-panel loops to a vertical orientation as shown here.</p>
---	--

The following table shows the excess noise that is **required** at various frequencies. These values assume the front-panel loops have been re-configured as shown above:

Frequency range	Required Excess Noise Power
Up to 20 GHz:	30 dB
Up to 50 GHz:	40 dB
Up to 67 GHz:	45 dB

For devices that do NOT meet this requirement, a low-noise amplifier (LNA) must be added to the receiver loop (see following image). This boosts the noise power at the receiver by the gain of the LNA. The disadvantage is the possibility of measurement drift and receiver compression. Any change in the gain of the LNA will have an impact on measurements that use the receiver with the LNA, so frequent calibration may be required. Care should also be taken when setting the channel power (used during the S-parameter portion of the calibration, and the gain portion of the DUT measurement) to ensure that the added gain of the LNA does not cause receiver damage or compression. A filter is also required on the output of the LNA. [Learn more.](#)



Filtering Requirement (Option 028)

Opt 029 includes noise receivers with filtering to keep mixing-product noise out of the low-noise receivers. These filters are not available when measuring noise with the standard PNA receiver. Therefore, for best measurement accuracy, a filter should be used at the output of the DUT (or LNA preamp if used).

- A bandpass filter at the frequencies of interest can always be used.
- A lowpass filter can be used when the PNA is doing fundamental mixing (up to 26.5 GHz). The lowpass filter must pass the fundamental frequency of the measurement but suppress the third harmonic. A measurement at 1 GHz would need a lowpass filter with a cutoff below 3 GHz, while a 5 GHz measurement would need a filter with a cutoff below 15 GHz.
- A single highpass filter can often be used when the PNA is doing 3rd-harmonic mixing (from 26.5 to 67 GHz). Use a highpass filter with cutoff about 18 GHz for frequencies up to 50 GHz. For operation to 67 GHz, the filter cutoff would need to be above 23 GHz.

Using Noise Figure Traces in Equation Editor

In a [Noise Power](#) trace, the underlying unit is noise temperature.

```
10*log10( temperature * 1000mw/w * 1.38e-23)
```

(**1.38e-23** is Boltzmanns constant)

Any time you use Equation Editor on a Noise Power trace, the LogMag formatting will apply the above equation. Therefore, first select **REAL** format and then generate the equation.

Radio-Frequency Electromagnetic Field Immunity

When a 3Vm-1 radio-frequency electromagnetic field is applied to an PNA with Opt 029 according to IEC 61000-4-3:1995, degradation of performance may be observed. When the frequency of the incident field matches the frequency of a measured noise figure or gain, the values displayed will deviate from those expected. This phenomenon will only affect that specific frequency, and the analyzer will continue to perform to the specification at all other frequency sample points.

The PNA with Opt 029 may be unable to calibrate a chosen frequency sample point if the frequency matches that of an incident electromagnetic field.

Last Modified:

17 June 2013	Added supported Noise Tuners
29-Mar-2013	Added link to config guide
30-Jan-2013	Added Save Data
21-Dec-2012	Added cable loss to the Gain+NF equation
24-Oct-2012	Add Noise Parameters
10-Oct-2012	Modified 50 GHz (2 places)
5-Sep-2012	Added 50 GHz models
3-Aug-2012	Added auto BW adjust
21-May-2012	Added Power sensor cal
1-Mar-2012	Edits to 028 tips (DB)
28-Feb-2012	Modified Scalar TRL link
14-Dec-2011	Replaced config for LNA #2
4-May-2011	Edited for N522x
27-Apr-2011	Removed Copy Channels limitation
7-Apr-2011	Edited cal bandwidth and average
11-Mar-2011	Added note for Noise Sources
3-Dec-2010	Clarified Noise BW setting
2-Sep-2010	Added using equation editor
28-Apr-2010	Added port mapping (9.22)
12-Mar-2010	Added 028 and H29 - removed some fixturing limitations (9.20)
19-Nov-2009	Added X-axis point spacing (9.1)
31-Jul-2009	Added interference tip
15-Jul-2009	Added reference to other tuner models
6-Apr-2009	Replaced N5242A with PNA
25-Feb-2009	Added Incident parameters and Scalar (A.08.50)
18-Feb-2009	Added Noise model
29-Sep-2008	Added Port Ext to NOT supported list
22-Apr-2008	Added link to preferences
12-Mar-2008	Modified Noise Params
29-Jun-2007	MX New topic

Noise Figure on Converters (NFX) - Opt 028 and Opt 029

This topic discusses Noise Figure on Converters:

- [Requirements and Limitations](#)
- [How to Configure your Hardware](#)
- [Create a NFX Measurement](#)
- [NFX Parameters](#)
- [Valid Mixer Configuration / Sweep Type Combinations](#)
 - [Frequency tab](#)
 - [Power tab](#)
 - [Noise Figure Setup tab](#)
 - [Mixer Frequency tab](#) (separate topic)
 - [Mixer Power tab](#) (separate topic)
 - [Mixer Setup tab](#) (separate topic)

The following **general** information (contained in [Noise Figure Application](#) for Amplifiers) is also relevant for NFX measurements:

- [Noise Figure Options Explained \(029, 028, H29\)](#)
- [Features and Requirements](#)
- [Noise Concepts](#)
- [How the Noise Figure Application Works](#)
- [Scalar Noise Figure Measurements](#)
- [Perform Calibration](#)
- [Noise Figure Measurement Tips](#)
- [Noise Model and the Noise Correlation Matrix](#)

Other Application Topics

Requirements and Limitations

Noise Figure on Converters requires [Noise Figure](#) (Option Opt 028 or 029) and [FCA](#) (Opt 082 or 083).

[Learn more about Noise Figure requirements.](#)

Limitations

- Upconverters OR downconverters ONLY (not both).
- Image rejection is NOT provided in NFX.

The following PNA features are **NOT** available with NFX:

- Analog Sweep ([Stepped sweep](#) mode only)
- Independent IFBW, Power Levels, or Sweep Time in a [segment table](#) is NOT supported.
- [Log frequency](#) sweeps
- [ECal User Characterization](#)
- [Time Domain](#)
- [Balanced measurements](#)
- Save [Formatted Citifile](#) data.
- [Interface Control](#)
- Some [Fixturing Features](#)
- [External Test Set Control](#) (Option 551)
- [External DC Devices](#)
- [Pulse measurements](#)
- [See Frequency Limitations](#)

Embedded LO measurements **ARE** supported in NFX. [Learn more.](#)

How to Configure your Hardware

The PNA-X is extremely versatile, and can be configured in many ways to make NFX measurements. While not all conceivable configurations are documented here, a few of the most common examples are provided to show the basic concepts.

DUT Configuration

[Learn more about connecting the DUT input and output to the PNA.](#)

The DUT LO can be connected to an external source OR [PNA internal second source \(if available\)](#).

Select an LO Source on the [Mixer Setup tab](#).

Note: Noise that is present on the LO source will be directly transferred to the DUT output. This noise can NOT be calibrated out of the noise measurement. Therefore, choose a low-noise source for the LO, such as an Agilent ESG or PSG, which is better than the PNA internal source.

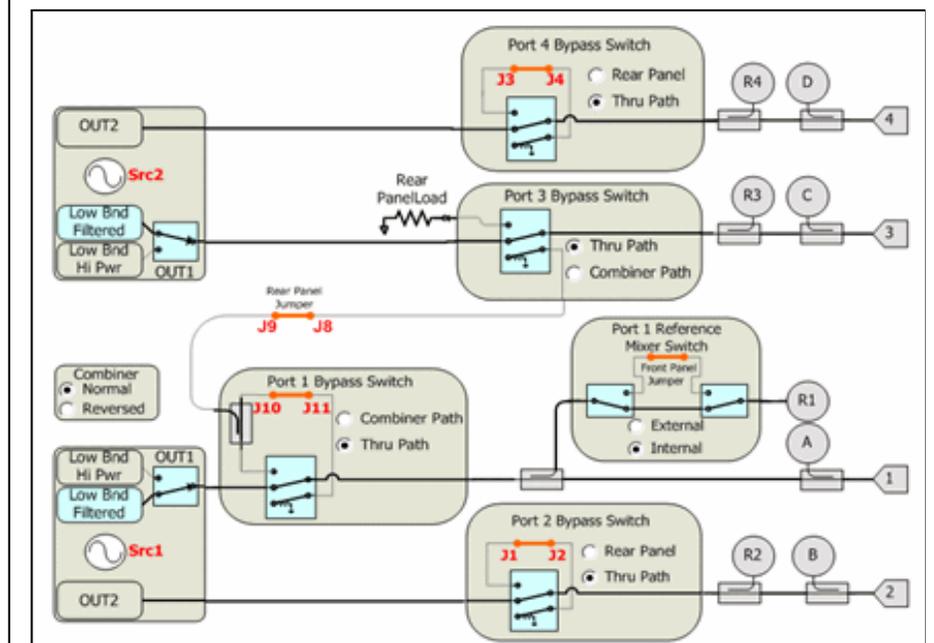
External LO Source through Port 3 or Port 4 (4-port PNA-X only)

- Connect the DUT LO to PNA-X Port 3 or Port 4.
- Connect external source to rear-panel J7 (**Rear-panel load** on below diagram) for Port 3; J3 for Port 4.
 - For Port 3, NO switching is required.
 - For Port 4, switch Port 4 Bypass Switch to **Rear Panel**.

PNA-X Internal LO Source

- SRC 2 **Out 1** or **Out 2** on front-panel or 2-port / 2-source PNA-X
- **Port 3** or **Port 4** on 4-port PNA-X

PNA Source 2 for LO on 4-port models



Path Configurator - Default configuration on PNA-X [option 423](#)

- **By default, Internal source2 (Src2) is supplied at ports 3 and 4.**
- **Bypass switches must be in "Thru Path"**

Using the Noise Figure Application

Use the following general procedure to make measurement with the Noise Figure App:

1. [Connect Tuner and Noise Source.](#)
2. [Create a Noise Figure Measurement.](#)
3. [Make Noise Figure Settings.](#)
4. For Opt 029 and H29, copy your Noise Source ENR file to the PNA C:/Program Files/Agilent/Network Analyzer/Noise folder.
5. [Perform Calibration](#)
6. Connect the DUT. [Learn more.](#)
7. Measure Noise Figure.
8. **Optional** Click File, then Save to save Noise Figure data in the following [formats](#): (available ONLY when NF correction is ON.)
 - ***.CTI** Citifile
 - ***.PRN**
 - ***.nco** Noise Correlation Matrix data in S2P format. [See Noise Model.](#)

See Also: [Measurement Tips](#)

Connect Noise Tuner and Noise Source

1. Connect the **noise source** to the [28V connector](#) on the PNA-X rear panel. **NOT required for Opt 028.** The Noise Source is turned ON and OFF automatically as needed during a calibration. Connect the noise tuner to Port 2 reference place when prompted during calibration.
2. Connect the **noise tuner** (ECal module). NOT required for [50 GHz models](#) and [Scalar Noise Figure](#) measurements.
 - a. On the PNA front panel, remove the **Port 1** jumper cable SOURCE OUT / CPLR THRU. Opt 028 allows Noise Figure measurements using any two PNA ports. Connect the noise tuner to the front-panel jumpers for the source (DUT input) port.

- b. Connect M-F tuner (N4691B-M0F) using the supplied cable (N5242-20137) and adapter (85052-60013).
- c. When using F-F ECal module (N4691B-00F), order a 3.5 mm M-M adapter (85052-60014).



Create a Noise Figure Measurement

1. On the PNA front panel, press **Meas**, then **[Measurement Class]**
2. Select **Noise Figure Converters**, then either:
 - **OK** delete the existing measurement, or
 - **New Channel** to create the measurement in a new channel.
3. A Noise Figure measurement is displayed. Do the following to add or change parameters to display.

NFX Parameters

How to ADD NFX Parameters

1. Create an NFX channel.
2. Then do the following:

Using front-panel HARDKEY [softkey] buttons

1. Press **TRACES**
2. then **[New Trace]**
3. then select a parameter

Using a mouse with PNA Menus

1. Click **Trace/Chan**
2. then **New Trace**
3. then select a parameter

How to CHANGE NFX Parameters

1. Create an NFX channel.
2. Select the parameter to change.
3. Then do the following:

1. Press **MEAS**
2. then select a parameter

1. Click **Response**
2. then **Measure**
3. then select a parameter

Programming Commands

The same Noise parameters that are available in the Noise Figure application are also available in the NFX application. [Learn more.](#)

In addition, the following Mixer and Raw Receiver parameters are available in an NFX channel:

Mixer Parameters

- **SC21**- Conversion Loss
- **SC12** - Reverse Conversion Loss
- **S11** - Input match
- **S22** - Output match
- **IPWR** - Input power to mixer/convertor. Same as R1 (Source1)
- **OPWR** - Output power to mixer/convertor. Same as B (Source1)
- **RevIPWR** - Power applied to mixer/convertor Output. Same as R2 (Source2)
- **RevOPWR** - Power measured at mixer/convertor Input. Same as A (Source2)

Raw Receiver Parameters

Specify a receiver to measure at LO1 frequencies with the notation:

- <Receiver>LO1
- For example: **ALO1** or **R1LO1**

Specify a receiver to measure using a source port, with the notation:

- <Receiver>_<source port>
- For example: **A_3** or **R1_1**

How to start the Noise Figure Setup dialog

To provide quicker access, use the Setup softkey. [Learn how.](#)

Using front-panel HARDKEY [softkey] buttons

1. Press **FREQ**
2. then **[Noise Figure Setup]**

Using a mouse with PNA Menus

1. Click **Response**
2. then **Measure**
3. then **Noise Setup**

Programming Commands

Valid Mixer Configuration / Sweep Type Combinations

The following are the **Valid Mixer Configurations**:

Sweep Type	Input	LO	Output
Linear	Swept	Swept	Fixed
	Swept	Fixed	Swept
CW	Fixed	Fixed	Fixed
	Fixed	Swept	Swept

For determining a valid mixer configuration with 2 LOs, one Fixed LO and one Swept is equivalent to having a single-stage Swept LO.

If you create an invalid Sweep Type / Mixer Configuration, a red message appears like the following:

ERROR: Input range must be swept in Linear sweep mode

If this occurs, change the **Sweep Type** on the [Frequency](#) tab.

[See other rules for configuring a mixer.](#)

The following tabs are available on the NFX Setup dialog:

- [Frequency](#)
- [Power](#)
- [Noise Figure Setup](#)

- [Mixer Frequency](#) (separate topic)
- [Mixer Power](#) (separate topic)
- [Mixer Setup](#) (separate topic)

Frequency tab - NFX Setup dialog box help

The screenshot shows the NFX Setup dialog box with the following sections:

- Sweep Type:** Radio buttons for "Linear Sweep" (selected) and "CW Frequency".
- X-Axis Display:** A dropdown menu labeled "Annotation:" with "Output" selected.
- Sweep Settings:** A grid of input fields:
 - Number Of Points: 201
 - IF Bandwidth: 1.000 kHz
 - Start: 800.000000 MHz
 - Stop: 300.000000 MHz
 - Center: 550.000000 MHz
 - Span: 500.000000 MHz

Sweep Type

Linear - Use for Swept Input parameters

CW Frequency - Use for Fixed Input parameters

X-Axis Display

Annotation: Select the frequency range to display on the X-axis.

Sweep Settings

Click to learn more about these settings.

- [Number of points](#)
- [IF Bandwidth](#) For standard PNA receiver measurements. This setting is important for improving noise measurement accuracy. [Learn more.](#)
- [Start / Stop](#), [Center / Span](#) frequencies.

[Learn about the Load/Save *.mxr files](#), and other buttons across the bottom of all NFX Setup tabs.

Power Tab - NFX Setup dialog box help

Power On (All Channels)

DUT Input Port

Input Port: Port 1

Power Level: -30.00 dBm

Source Attenuator: Auto 20 dB

Receiver Attenuator(A): 0 dB

Source Leveling Mode: Internal

DUT Output Port

Output Port: Port 2

Power Level: -20.00 dBm

Source Attenuator: Auto 0 dB

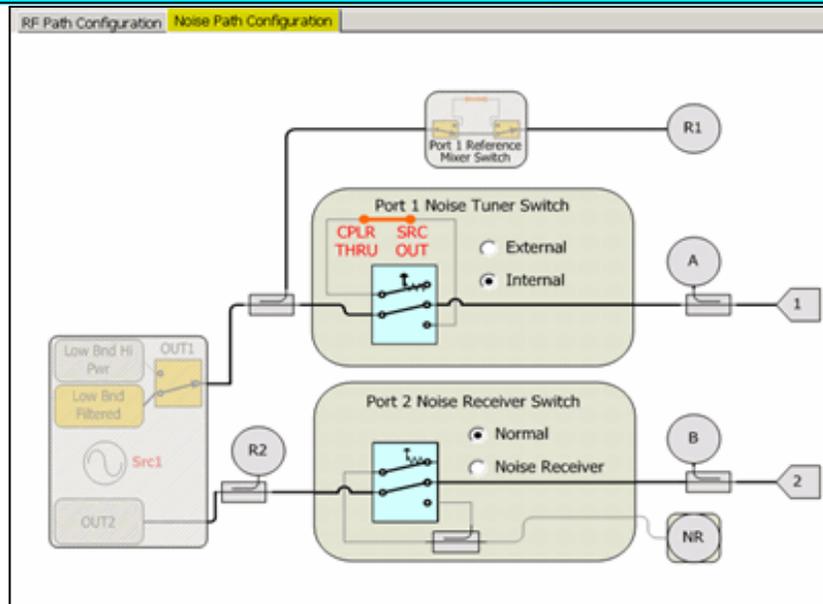
Receiver Attenuator(B): 0 dB

Source Leveling Mode: Internal

[Learn about this dialog.](#)

[Learn about the Load/Save *.mxr files](#), and other buttons across the bottom of all NFX Setup tabs.

Noise Path Configurator dialog box help



[Learn about this and the Path Configuration dialog for the 50 GHz models.](#)

NFx Noise Figure Setup dialog box help

Bandwidth/Average
Noise Bandwidth: 4.0 MHz
Average Number: 1
 Average ON

Noise Receiver
 NA Receiver (Port 2)
 Noise Receiver

Receiver Gain
 High
 Medium
 Low

Ambient Temperature
295.00 K

Impedance States
Noise Tuner: None selected
Max Acquired Impedance States: 4
 Enable Source Pulling for SParameters (For mixers with low reverse isolation)

This dialog is identical to the Noise Figure Setup for amplifier EXCEPT for the following setting. [Learn about the rest of this dialog.](#)

Enable Source Pulling for S-Parameters (For mixers with low reverse isolation). When checked, during S22 (output match) measurements, the noise tuner is switched to present different impedance states to the DUT input. From these measurements, S22 is computed as though the input is seeing a 50 ohm match. This requires more sweeps. Check this box when the converter has low reverse isolation, as is the case when the NO output path is NOT padded with attenuation. Otherwise, clear this checkbox as S22 measurements will not be improved.

An accurate S22 measurement is essential when measuring S-parameters during an NFX calibration.

[Learn about the Load/Save *.mxr files](#), and other buttons across the bottom of all NFX Setup tabs.

Mixer Frequency tab - NFX Setup -dialog box help

Mixer Frequency

Input: Start/Stop | 1.00000000 GHz | 2.00000000 GHz | Calc Input

LO1: Fixed | 500.000000 MHz | Input > LO

IF: Start/Stop | + | 1.50000000 GHz | 2.50000000 GHz | Calc LO1
 - | 500.000000 MHz | 1.50000000 GHz | Calc LO2

LO2: Fixed | 500.000000 MHz | IF1 > LO2

Output: Start/Stop | + | 2.00000000 GHz | 3.00000000 GHz | Calc Output
 - | 1.00000000 GHz | 2.00000000 GHz

[Learn about this dialog.](#)

Mixer (LO) Power tab - NFX Setup dialog box help

Power On (All Channels)

LO1: Not Controlled
LO1 Power: -10.00 dBm
Source Leveling Mode: [Dropdown]

LO2: Not Controlled
LO2 Power: -10.00 dBm
Source Leveling Mode: [Dropdown]

Port Settings

Port 3
Source Attenuator: 5 dB
Receiver Attenuator: 0 dB

Port 4
Source Attenuator: 5 dB
Receiver Attenuator: 0 dB

Swept Power Settings

	Start	Stop	Step
LO1 Swept Power:	-10.00 dBm	0.00 dBm	0.050 dB
LO2 Swept Power:	-10.00 dBm	0.00 dBm	0.050 dB

Path Configuration...

[Learn about this dialog.](#)

Mixer Setup tab - NFX Setup dialog box help

Converter Stages: 2

Hardware Configuration
Port 3: Thru
Port 4: Thru

Converter Model: Single Stage

Port 1 → [Mixer] → [Multiplier] → [Multiplier] → Port 2

LO1: Not controlled
LO2: Not controlled

[Learn about this dialog.](#)

Last Modified:

- 6-Nov-2012 Added noise parameters
- 31-Oct-2012 Added DC Devices limitation
- 30-Jul-2012 Small edits
- 27-Apr-2011 Removed Copy Channels limitation
- 16-Aug-2010 Divert mixer tabs
- 30-Apr-2010 Added 028 and port mapping
- 3-Mar-2010 Fixed Meas Class - removed some fixturing limitations
- 6-Oct-2009 MX New topic (A.09.10)

Calibration for Noise Figure on Amplifiers and Converters (NFX)

This topic discusses calibration for both Noise Figure on Amplifiers and Noise Figure on Converters (NFX).

- [Overview](#)
- [How to Perform a Noise Figure Cal](#)
 - [Select Calibration Method](#)
 - [Configure Noise Source](#)
 - [Select DUT Connectors and Cal Kits](#)
 - [Measure Standards Steps](#)
 - [NFX Calibration](#)
 - [Validate Noise Source Cal](#)

See Also

[Noise Figure and TRL Cal](#)

See [Noise Figure and NFX Applications](#)

Noise Figure Calibration Overview

Note: Noise Figure results are NOT at all accurate without a Noise Figure calibration.

NFX Calibration

Noise Figure calibration is very similar for both amplifiers and converters (NFX).

- NFX does NOT offer '[S-params ONLY](#)' calibration.
- NFX includes an optional [LO Power Cal](#).
- NFX Cal may generate a Cal Kit Frequency Error message. [Learn more](#).

Calibrating the Noise Receivers

Note: The term 'noise receiver' is used here to refer to the receiver that is used to measure noise. It can be a standard PNA receiver or the dedicated noise receivers that are provided with Opt. 029.

The noise figure calibration process is different depending on if a Noise Source or a PNA source (calibrated with a power meter) is used to calibrate the noise receiver.

Using a Power Meter

When 'Use Power Meter' is selected on the '[Select Cal Method](#)' dialog, a power meter is used in place of a noise source to characterize the noise receiver. The process happens in three steps:

1. A Source Power Cal is performed at the port connected to the DUT's input, with a power level that is specified on the [first measurement step](#) of the calibration wizard.
2. A THRU connection is made from the calibrated source port to the specified noise receiver port. The gain of the noise receiver is then measured, as well as the receiver's noise floor.
3. With the THRU connection in place, the swept-frequency response of the noise bandwidth filter is measured. Since the noise receiver uses double-sideband homodyne mixing, the user sees a symmetrical response representing the low- and high-side responses, with a notch in the middle that nulls out the DC response. From the measured filter shape, the equivalent noise bandwidth is calculated. This information combined with the data from step 2 gives the gain-bandwidth product and noise figure of the noise receiver.

The following are variations to this process depending on the type of noise receiver selected.

Using the low-noise (Opt 029) receiver:

- a. Step 3 (measure the frequency response of the noise bandwidth filter) is always performed, over the frequency range specified in the noise figure channel.
- b. The gain-bandwidth information is contained within the noise figure calset.
- c. **Note:** 'Use Power Meter' is NOT allowed when the [Noise bandwidth](#) is set to 8 MHz or 24 MHz.
- d. Noise averaging is not automatically turned on.

Using a standard receiver:

- a. Step 3 (measure the frequency response of the noise bandwidth filter) is performed one time for each noise bandwidth and each measurement receiver. When this step occurs, it happens over the full frequency range of the PNA. Therefore, a broadband connection must be ensured -- there should be NO filtering in the thru path. For example, a wave-guide-to-coax adapter, which has a high-pass frequency response, should NOT be included.
- b. After the initial cal, the noise bandwidth data is saved to an xml file and reused for subsequent noise calibrations. As new noise bandwidths and receivers are selected, new data is appended to the xml file. If you suspect that the xml file might contain bad values for the noise filter bandwidths, then delete:
For XP - C:\Program Files\Agilent\Network Analyzer\Noise\noiseBW.xml
For Win7 - C:\programdata\agilent\network analyzer\Noise\noiseBW.xml
This will cause the noise-filter shape for the selected receiver to be remeasured the next time a noise cal is performed using a standard receiver.

- c. Noise averaging is automatically turned on, with a default value of 100 noise averages. This value can be modified as necessary. [Learn more about Noise Averaging.](#)

Using a Noise Source ([See Noise Source requirements](#)). NOT used when measuring noise figure with Standard PNA receivers.

A Noise Source is a device that generates two very consistent levels of noise over its operating frequency range:

- Hot (On) - the Noise Source is biased in order to provide a high level of noise.
- Cold (Off) - the Noise Source is unbiased to provide ambient temperature noise level.

These levels are measured by the Noise Source manufacturer and provided in table and electronic format with each Noise Source by serial number. The electronic file is known as the ENR (Excess Noise Ratio) file.

1. The Noise Source is connected to the noise receiver through test port 2.

Note: For highest accuracy, the noise source should be connected as close as possible (the least amount of electrical loss) to the PNA port 2 connector. This causes the largest difference between the Noise Source HOT (on) and COLD (off) settings.

2. The Noise Source is measured by the noise receivers at each measurement frequency. The differences between the known ENR noise levels and the measured noise levels are the noise error terms. These values are removed from subsequent noise measurements.
3. During the Noise Source measurements, noise averaging and noise bandwidth is automatically turned ON to the values that you specify. [Learn more about Noise Averaging.](#)

Following the Noise Receiver Cal

- A **2-port S-parameter calibration** is performed on the noise figure channel. This is because S-parameters are measured at each frequency step before a noise measurement. Also during the S-parameter cal, at least FOUR different impedance states are presented at port 2 in order to later characterize the noise generated by the noise receiver. This cal can be either a SOLT or [TRL cal](#). See [Noise Figure and TRL Cal](#).
- After calibration, correction is automatically turned ON. The PNA [status bar](#) shows **VNC_2P** (for Vector) or **SMC_2P** (for Scalar).

How to Perform a Noise Figure Calibration

- Make the noise figure channel the active channel.
- Connect the noise figure Tuner to the PNA (for Vector noise figure cal).

Using front-panel HARDKEY [softkey] buttons

1. Press **CAL**
2. then [**Cal Wizard**]

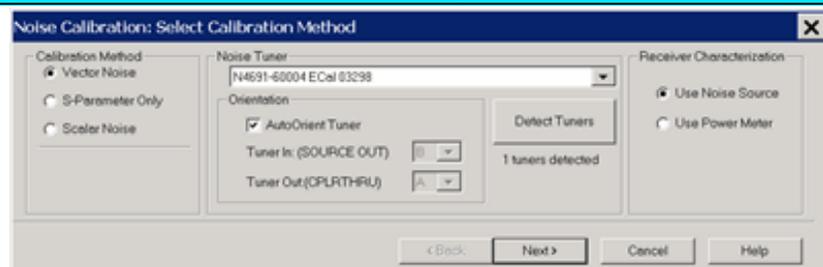
Using a mouse with PNA Menus

1. Click **Response**
2. then **Cal**

Programming Commands

The following Cal Wizard pages are unique to noise figure calibration. The remaining pages that are presented are the same as those in the standard [Cal Wizard SmartCal](#).

Select Calibration Method dialog box help



Calibration Method

- Vector Noise - Comprehensive noise figure calibration
- S-Parameter Only - Does NOT calibrate the noise receivers. NOT offered with NFX.
- Scalar Noise - Calibration for scalar noise figure measurements. [Learn more.](#)

Enable LO Power Cal - NFX ONLY. Check to cause the Cal Wizard to guide you through a Power Calibration on the LO source.

Note: NO correction is provided for an adapter that may be used to connect the power sensor to the LO source.

Noise Tuner

- Not available when Scalar Noise is selected.
- Select from the ECal modules that are connected to the USB.
- For 50 GHz Noise Receivers, select "Internal" to use the built-in Noise Tuner. [Learn more.](#)

Orientation

AutoOrient Tuner Check to allow the noise tuner orientation to be auto-detected. When cleared, use the following two fields to provide manual orientation of the noise tuner.

Tuner In (SOURCE OUT) / Tuner Out (CPLRTHRU): Specify the ECal module labels that are connected to the PNA front panel jumper connectors. [Learn how to connect the noise tuner.](#)

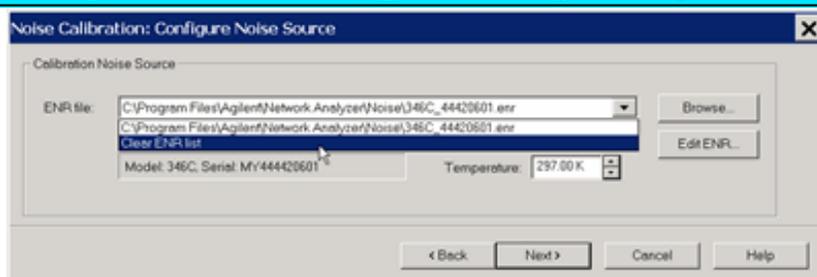
Detect Tuners Click to re-detect the Noise Tuners (ECal modules) that are connected to the USB. If the ECal module is not detected, check the USB connection, then click this button. The label below the button indicates the total number of ECal modules that are connected to the USB.

Receiver Characterization - [Learn more about this process.](#)

- **Use Noise Source** - A noise source is used to characterize the low-noise receivers.
- **Use Power Meter** - A Power Meter/Sensor is used to calibrate a PNA source, which then is used to characterize either the low-noise receivers or a PNA receiver. This selection is made for you and can NOT be changed when **NA Receiver** is selected on the [Noise Figure Setup dialog](#).

Note: Use Power Meter is NOT available when the Noise Bandwidth is 8 MHz or 24 MHz. In the [Noise Setup dialog](#), lower the Noise Bandwidth to allow this selection.

Configure Noise Source (Opt 029) dialog box help



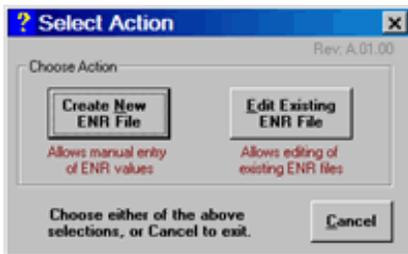
ENR File Select the Noise Source ENR file. If not already there, copy your Noise Source ENR file to the PNA C:/Program Files/Agilent/Network Analyzer/Noise folder. Then click Browse to find the ENR file.

Clear ENR List Scroll to the bottom of the ENR list, then click to removes the selected ENR file. Then browse or select to find a new file.

Edit ENR Click to launch the [ENR Editor](#) dialog box which is used to change or create ENR files. This is NOT usually necessary.

Temperature Specify the current temperature at the Noise Source connector. The Noise source is kept ON during noise figure measurements. This results in the Noise Source being 4 to 5 degrees warmer than Ambient temperature, and a more accurate calibration. See [Noise Figure tips](#) to learn more about the significance of temperature.

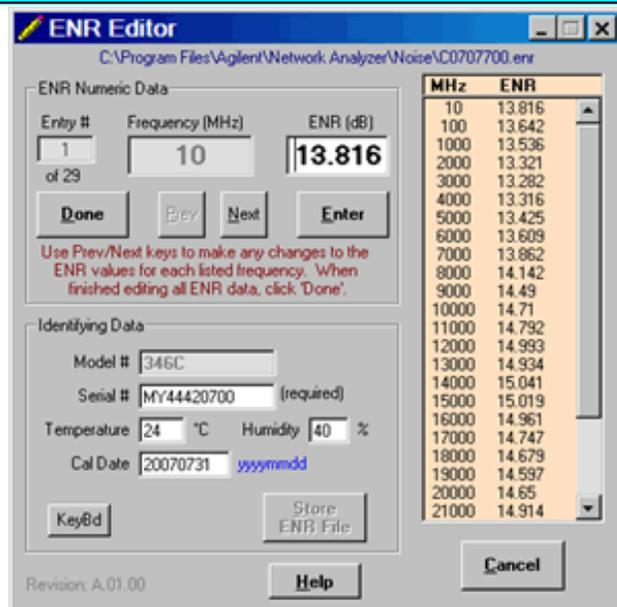
[See Noise Source requirements.](#)



Click either Create or Edit to launch the same dialog box, shown below.

- **Edit** populates all fields with existing data which can then be edited and stored.
- **Create** has empty fields except for frequencies.

Edit / Create ENR File dialog box help



ENR Numeric Data

Use **Previous** and **Next** buttons to scroll to **Entry #** to edit. Type **ENR** value in dB, then press **Enter**.

Done Click when finished editing all values. Then click **Store ENR File** to save the file.

Identifying Data

Model # of the Noise Source. This can NOT be changed.

Serial # of the Noise Source.

Temperature and Humidity in which the Noise Source was calibrated. This is for information only. The ENR data is always normalized to 290 Kelvin.

KeyBd launches a mouse-driven keyboard.

Store ENR File Click to launch a dialog to save the new or edited ENR file.

Select DUT connectors and Cal Kits dialog box help



Port 1 and Port 2

DUT (Device Under Test) Connectors Specify the connector and gender of the **DUT**.

Cal Kits Select the Cal Kit to be used to calibrate each test port. The list for each DUT Port displays kits having the same connector type as the DUT. Using incorrect calibration standards can significantly degrade measurement accuracy. [Learn more.](#)

Power Sensor Used to calibrate the source port. Specify the connector and gender of the Power Sensor.

Noise Src Used to calibrate the noise receivers (Opt 029). The Agilent 346C has an "APC 3.5 male" connector.

Note: For highest accuracy, the noise source should be connected as close as possible to the PNA port 2 connector. This causes the largest difference between the Noise Source **HOT** (on) and **COLD** (off) settings.

For both Cal devices (power sensor and noise source, specify the connector type and gender. When the Cal device connector is **NOT** the same type and gender as the DUT Port connector, then for optimum accuracy, extra cal steps are used to measure and correct for the adapter that is used to connect the Cal device to the reference plane.

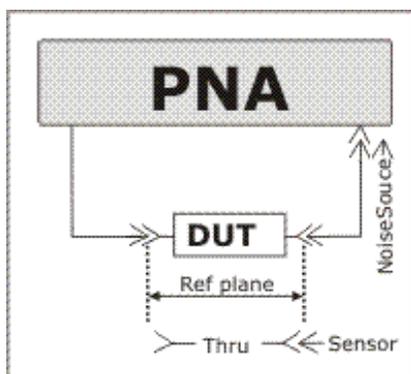
Select **Ignored** (at the bottom of the DUT Connectors list) to NOT compensate for the adapter.

Select the Cal Kit that will be used for that process.

De-embed power sensor adapter / noise source adapter The PNA uses the connector type and gender of the DUT along with the connector type and gender of the cal device to determine if an adapter removal operation is taking place AND whether or not that removal operation requires an additional cal step.

However, the use of the connector type can, in special cases, hide the need for the extra cal step. Check the "De-embed..." box in these cases to inform the PNA that the extra step is needed.

Such a case is illustrated below where the noise source is connected close to test port 2 for higher accuracy. If unchecked, the PNA would assume in this case that the Noise Source is connected to the Thru standard at the port 1 (DUT input) reference plane.



Source Cal Settings Click to launch the [Source Power Cal \(for apps\)](#) dialog. This dialog is used to set Power Meter / Sensor settings for both the Port 1 Power Cal, and the optional LO Power Cal.

Modify Cal Check, then click **Next**, to Modify Cal (Standards AND Thru Method).

Note: [Enhanced Response Calibration](#) is NOT supported with noise figure.

Measure Standards Steps dialog box help



Power Level at which to perform the Power Cal.

It is usually best to set power level to 0 dBm at the power sensor because the power sensor is calibrated at that level. Lower power levels will yield a slower and noisier calibration.

However, with 20 dB of source attenuation (default NF setting), the PNA may not be capable of achieving this power level at higher frequencies. To check the max leveled power, view an [R1 \(port 1 reference receiver\) trace](#) over the frequency range of interest, then increase the power until roll-off appears. Power levels at the test port may be approximately 2 dB lower than at the R1 receiver.

If an external component is used between the PNA test port and the calibration reference plane, then adjust the power level so that the power at the sensor is about 0 dBm if possible.

The current source attenuation value is shown on the dialog.

LO Power Cal (Optional) When [enabled](#), perform a Source Power Cal at the DUT LO input. An LO must already be selected. [Learn how](#). The power level of the LO source calibration is set on the [NFX \(LO\) Power Tab](#).

Connect Noise Source to the Port 2 measurement (reference) plane

When the ["De-embed Adapter.."](#) boxes are checked, additional cal steps are required.

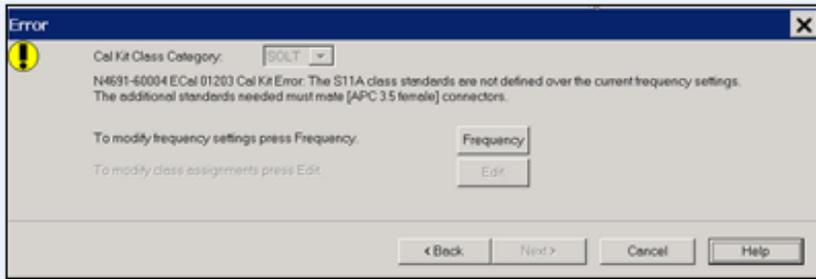
Subsequent Steps

Connect Port 1 to Port 2 - Connect port 1 reference plane to the port 2 reference plane using the required Thru standard or adapter.

Connect ECal to Ports 1 and 2 - Connect the ECal module between the port 1 reference plane and the port 2 reference plane.

NFX Cal Kit Frequency Error

When performing an NFX calibration with at least two ECal modules with different connector types, you may see the following error message.



This occurs because the PNA performs two full 2-port calcs: one at the input frequencies with both connector types, and one at the output frequencies with both connector types. One of the ECal modules may not be defined over both frequency ranges.

To overcome this, you can perform a User-Characterization for the limited ECal module over the required frequency range. The PNA will present a warning, but it will be allowed. [Learn how](#).

Validate Noise Source Cal

To validate a Noise Source calibration, connect the Noise Source to Port 2 and measure [ENR](#).

Compare the measured values to the values in the ENR table.

How to manually turn the Noise Source ON | OFF

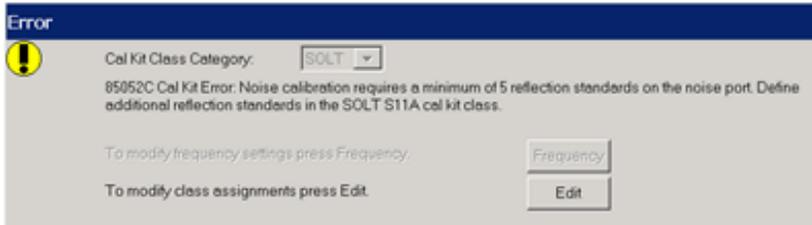
1. Press **POWER**
2. then [**Noise Source ON | OFF**]

Last Modified:

10-Sep-2013	Modified Power Sensor Cal (DB)
4-Apr-2013	Added link to TRL Cal
7-Sep-2012	Modified for 50 GHz
28-Feb-2012	Changed Scalar TRL
9-Jun-2011	Added Power Meter cal
16-Sep-2010	Added Scalar Noise and TRL Cal
30-Apr-2010	Added Opt 028 and port mapping
21-Oct-2009	Added NFX Cal
27-Feb-2009	Added Scalar (A.08.50)
9-May-2008	Added validation
23-Aug-2007	New topic

Noise Figure and TRL Cal

When performing a TRL (or LRL, LRM) Cal as the 2-port S-parameter calibration of a [Scalar or Vector Noise Figure](#) measurement, you may see an error message that states that there are not enough standards for the cal.



This appears because, during the TRL calibration, at least **FIVE** impedance states must be presented to the Noise Receiver port. A typical TRL Cal Kit does not have 5 standards with the same connector type and gender as the DUT output port, and with different impedances.

Note: Extra impedance standards are NOT required when you select and use an ECal module to perform the [De-embed noise source adapter](#). In this case the ECal module is used to present five different impedance states to the Noise Receiver port.

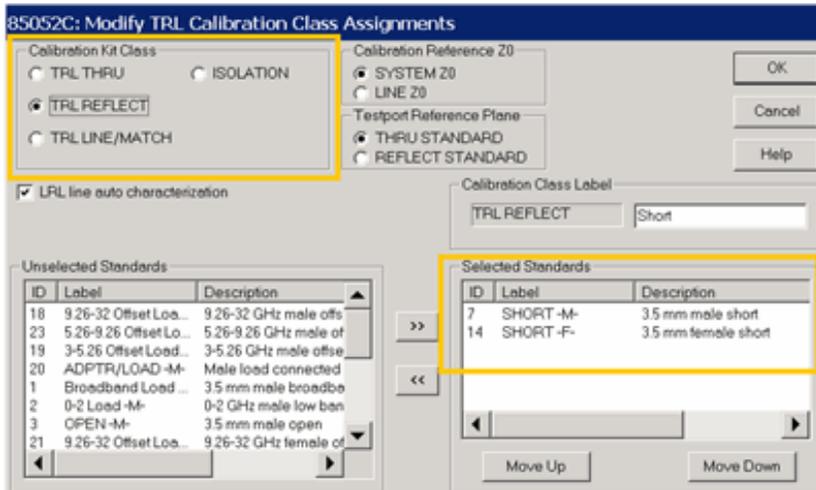
To correct this situation, you must define additional standards for your TRL Cal Kit using the [Modify Cal Assignments dialog](#).

You can view the impedance match by measuring the standard over the frequency range of interest while viewing the Smith Chart format. Ideally, all five standards should have a response at different areas of the [Smith Chart](#).

To Modify the Cal Kit

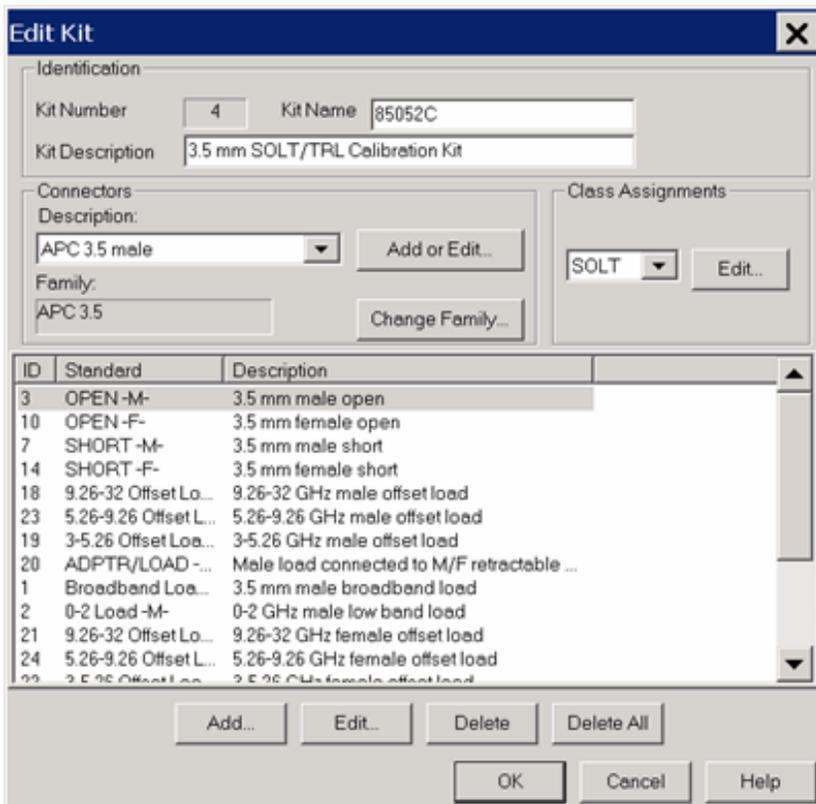
1. Click **Edit** on the Error dialog, or **Cancel** and start the Modify Cal Kits dialog.
2. For each of the following **Calibration Kit Classes**, note the **ID** number in the **Selected Standards** field:
 - TRL THRU
 - TRL RELFECT
 - TRL LINE/MATCH

For example, in the following image two shorts are defined as TRL RELFECT standards: ID numbers **7** and **14**. You can NOT use these IDs for the extra standards required by the Noise Cal. You must select other standards available in the kit or you can define new standards.

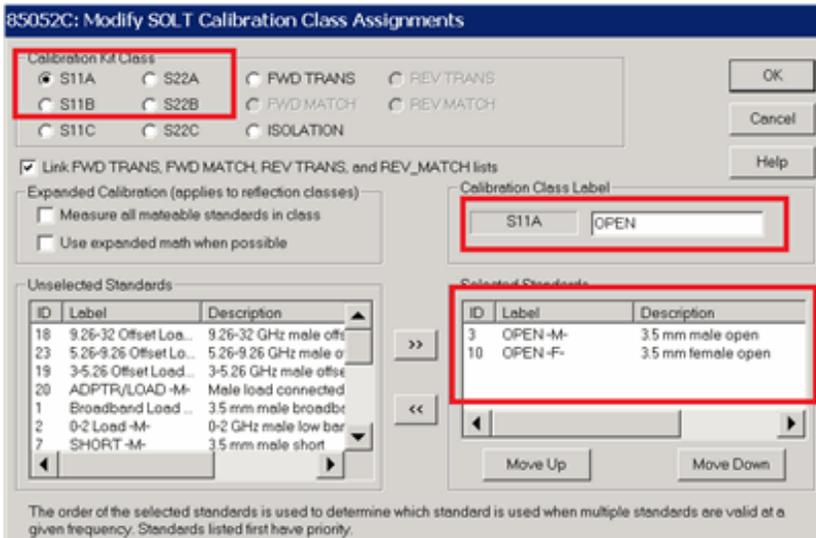


3. Select additional standards:

- Click **Cancel**.
- In the **Edit Kit** dialog, change **Class Assignments** to **SOLT**, then click **Edit** to start the **Modify Calibration Class Assignments** dialog.



4. In the following dialog, Select any of the **S11A**, **S11B**, **S22A**, **S22B** Calibration Kit Class definitions.



5. In the **Unselected Standards** field, find a standard that is NOT one of the TRL IDs noted above.
6. Click >> to add it to the **Selected Standards**.

Note: Be sure to choose standards with the **same** connector gender as the DUT output port.

The following are good options:

- Offset Short
- Offset Open

If your TRL Cal Kit does NOT have FIVE standards that meet these requirements, you can create a new device by reusing the line standard. However, instead of connecting both ends of the line, leave the input unterminated. The following is an example of how to create this device.

- Click **Cancel** to show the **Edit Kit** dialog.
- Click **Add**.

Thru/Line/Adapter

Identification

Standard ID: 5 Label: THRU

Thru Description: Unterminated LINE

Frequency Range

Min: 0 MHz

Max: 38800 MHz

Delay Characteristics

Delay: 0 pSec Loss: 0 Gohms/s

20: 50 ohms

Connectors

Port: APC 3.5 female Port: APC 3.5 male

Clear OK Cancel Apply Help

A new Standard ID is generated.

Change the Description to **Unterminated Line**. This provides a prompt to remind you to leave the line input unterminated when you connect it to the noise port.

Last Modified:

28-Feb-2012 Changed from Scalar TRL
11-Nov-2010 MX New topic

Narrowband Pulsed Application

Note: The [Integrated Pulse App](#) (Opt 008) replaces this application.

The Narrowband Pulsed Application is a Visual Basic program that provides a user interface for making pulsed measurements.

In this topic:

- [Required Options](#)
- [Connecting External Pulse Generators](#)
- [Using the Narrowband Pulsed Application](#)
- [How to Configure Pulse Generators and Receivers](#)
- [Calibration in Pulse Mode](#)
- [Pulse Profiling](#)
- [Signal Reduction versus Gate Width](#)
- [Pulsed Frequency Converter Measurements](#)
- [Writing your own Narrowband Pulsed Application](#)

The following enhancements were made In PNA Rev. 7.2:

- [Enhanced Pulse Measurement Capabilities](#)
- [Support for Internal Pulse Generators / Modulators \(PNA-X only\)](#)

See Also

- Learn about the [Wideband Pulsed Application](#).
- For more conceptual information see our [Pulsed Measurement App Notes](#).
- [See PNA-X Block Diagram of IF Path / Pulse Generators / Source Modulation](#)
- [Programming commands](#)

Other IF Access Topics

Required Options and Equipment

The PNA H08 option provides the Narrowband Pulsed Application. The following options are also required. If your PNA does not have the required options, a message is displayed on the screen. For more information, see [Pulsed-RF Measurements Configuration Guide](#)

Models

- PNA-X models: [Opts 021, 022, and 025](#) greatly enhance speed, performance, and convenience.
- PNA-X Opt 224 (2-port Dual Source) select the **2-port Dual Source** Configuration in the [Path Configuration](#) dialog to provide Pulse modulation on both port 1 and port 2.
- Use the [Pulse I/O connector](#) to access the internal pulse generators
- See the PNA-X [IF Path Configuration block diagram](#) which includes the Pulse Modulators and Generators.

Other PNA Models

- PNA-L models: H08 NOT available
- Agilent 81104A or 81110A Pulse Generator with ONLY the 81105A or 81111A output modules. The 81112A module does NOT have selectable 50 ohm/1K ohm output impedance/load compensation to drive the 1K ohm PNA IF gates. For more information, see the 81100 Family of Pulse Pattern Generators Technical Specifications at: <http://cp.literature.agilent.com/litweb/pdf/5980-1215E.pdf>

Connecting External Pulse Generators to the Z5623A H81

Each 81110A Pulse Generator has two output modules. Each output can drive a PNA IF Receiver or Source Modulation (Z5623A H81). Connect the Pulse Generators as follows:

81110A front panel connectors



- Connect GPIB cables to the 81110A and PNA.
- Connect the PNA 10 MHz Ref Out to the 81110A 10 MHz IN.
- If using two 81110As for a total of 4 outputs, then connect the TRIGGER OUT of one to the EXT INPUT of the other 81110A.
- Connect the 81110A OUTPUTs to the PNA rear panel IF inputs to be gated. The outputs are mapped in the [Pulsed Generator Configuration](#) dialog box.

Connect the Z5623A H81 Pulse Test Set (optional) to the PNA front-panel port 1 loops as follows:

PNA	H81
Src Out	Source IN
CPLR THRU	CPLR THRU
RCVR R1 IN	RCVR R1 Out

See Also

- [PNA Front-panel loops](#)
- [PNA-X IF Connectors](#)
- [81110A Documentation](#)
- [Z5623A H81 Documentation](#)

Using the Narrowband Pulsed Application

How to start the Narrowband Pulsed Application

Using front-panel HARDKEY [softkey] buttons	Using a mouse with PNA Menus
<ol style="list-style-type: none"> 1. Press MACRO 2. then [Pulse] 	<ol style="list-style-type: none"> 1. Click Utility 2. then Macro 3. then Pulse



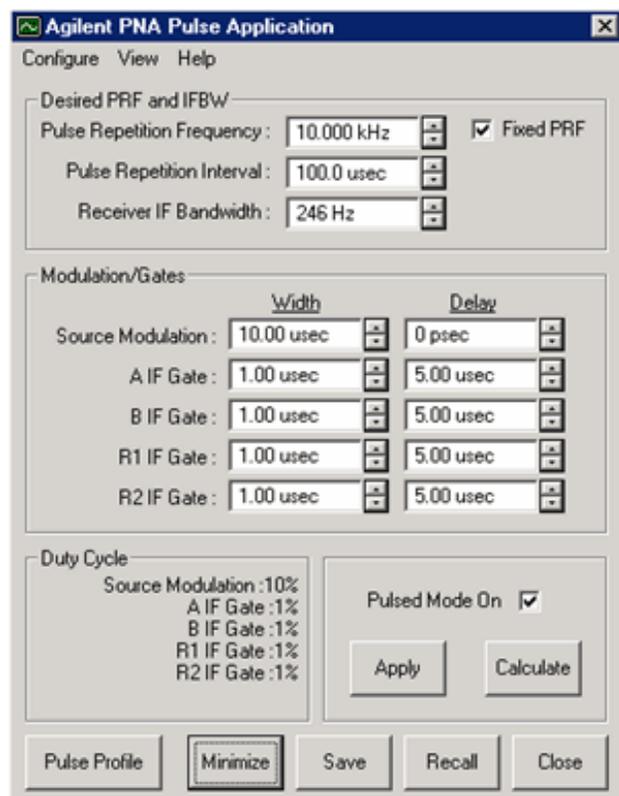
See Also

- [See programming commands to launch the Macro remotely.](#)
- [See how to write your own custom Narrowband Pulsed Application.](#)

Keypad Data Entry

The PNA front-panel Numeric Entry and Navigation keys can be used for dialog box input. Also, a keyboard can be used to enter values, including alpha characters for prefixes (for example, **u** for usec.) . After typing values, first press **Enter**, then press **Tab** to go to the next field.

The following is an image of the main dialog box:



Pulsed Application Main dialog box help

Note: An **error message** may appear on the PNA stating that the response frequency has exceeded the maximum allowed frequency.

The Narrowband Pulsed Application may set the offset frequency ([option 080](#)) of the PNA to some value other than zero (the default value). If the stop frequency is set to the maximum of the PNA model, then the error message will appear.

To fix this, set the stop frequency to a value that is at least 2 kHz less than the maximum allowed. For example, if you have a 20 GHz PNA, and the stop frequency is set to 20 GHz, and the error message appears, then set the stop frequency to 19.999998 GHz

[See Block Diagram of IF Path / Pulse Generators / Source Modulation](#)

Configure

You can configure more than one channel to make pulsed measurements, but the channels must use the same [pulse generator settings](#).

Only the Agilent 81110A Pulse Generator is supported with the Narrowband Pulsed Application. Refer to the 81110A documentation for pulse repetition frequency and duty cycle capabilities.

See Also

- [Configure Receivers](#)
- [Converter Measurements](#)

Edit / Undo Pulse Application settings revert to those when Apply was last pressed.

Desired PRF and IFBW Enter the DESIRED values. When **Calculate** is pressed, one or both of these values may change.

Pulse Repetition Frequency (PRF): Frequency of the pulses from the Pulse Generator.

Pulse Repetition Interval: $1/\text{PRF}$ Changes to either PRF or this setting changes both.

Receiver IF Bandwidth: IF Bandwidth of the PNA. Choose a setting from 1 Hz to 10 KHz.

Fixed PRF When checked, (default setting) the **Calculate** algorithm will NOT adjust the PRF, but only change the IF Bandwidth. When cleared, both the PRF and IF Bandwidth values are adjusted as necessary.

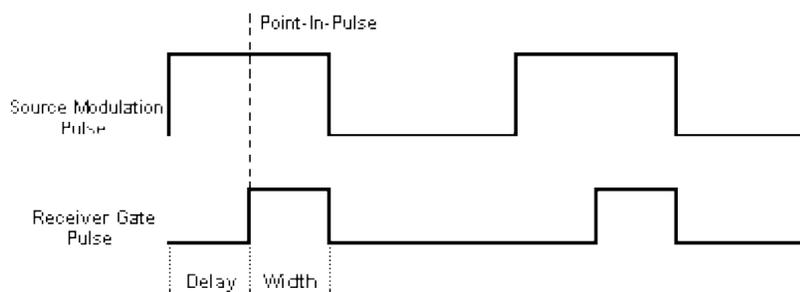
Note: On PNA's with [DSP version 4](#), the **Calculate** algorithm will NOT find nulling at several PRF frequencies. If this error is returned, add a small offset to the PRF (for example, 2.1 MHz instead of 2 MHz) or clear the **Fixed PRF** checkbox.

Modulation/Gates The Source Modulation and four PNA receiver gates can each have their own, or share, Pulse Generator outputs. Shared outputs have identical Width and Delay values. To configure and enable outputs, click **Configure**, then **Pulse Generators** to launch the [Pulsed Generator Configuration](#) dialog box.

Note: Option 036 and 037 limits the Source Modulation width to 117 ns.

Width Pulse Width.

Delay The delay that occurs before the pulse.



Duty Cycle Calculated Duty Cycle of the source and each of the selected receivers. Updated when **Calculate** is pressed.

Pulse Mode On When this box is checked, the PNA is enabled for Pulsed measurements. The PNA [Status Bar](#) annotation indicates the following:

- **G** Internal IF gates enabled.
- **F** Filtering for Pulsed Measurements enabled.

Apply All selections are sent to the pulse generator and the active channel of the PNA.

Calculate All selections are calculated and valid PRF and IFBW values are entered in their fields. If these settings are not acceptable, try changing the values you previously entered and click Calculate again. When acceptable values are attained, click **Apply** to send these values to the pulse generator and PNA.

Pulse Profile Launches the Pulse Profile dialog box. Same as clicking **View / Pulse Profile**. If not available, check **Pulse Mode ON**, click **Calculate**, then **Apply**.

Minimize Click to minimize the dialog box to make changes in the PNA application. To see the dialog again, select **Macro, Pulse**, or turn the [Status Bar](#) ON.

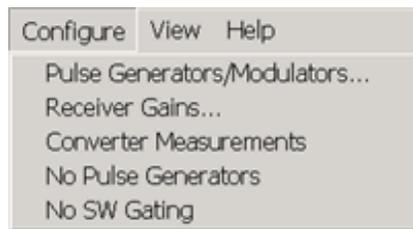
Save All settings from the Narrowband Pulsed Application are saved in a *.ppf file. These settings are NOT saved with PNA instrument state.

Recall Restore settings from the specified *.ppf file that were previously saved.

Close Closes the dialog box without saving changes.

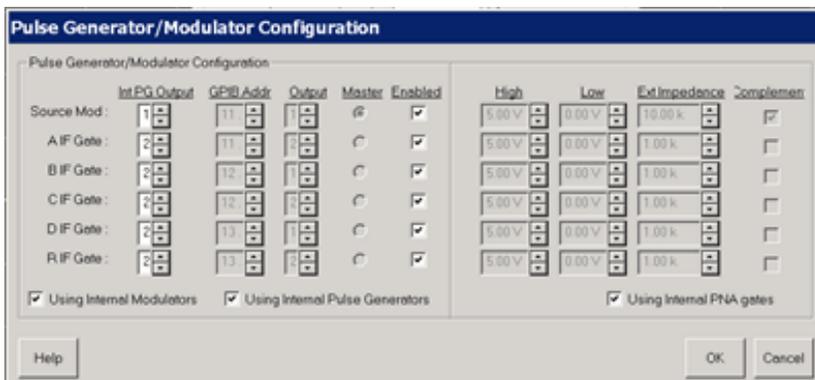
How to configure Pulse Generators / Modulators and Receivers

From the [Pulse App main dialog box](#)



Learn about...

- [Configure Receiver Gain](#)
- [Converter Measurements](#)
- **No Pulse Generators** When checked, the Narrowband Pulsed Application does NOT attempt to communicate with internal or external pulse generators. This setting is used for troubleshooting purposes.
- **No SW Gating** When checked, the improved SW gating sensitivity is turned OFF. This setting is used for troubleshooting purposes.



Pulsed Generator Configuration dialog box help

Notes:

- [See Block Diagram of PNA-X IF Path / Pulse Generators / Source Modulation](#)
- For PNA-X Opt 224 (2-port Dual Source) select the **2-port Dual Source** Configuration in the [Path Configuration](#) dialog to provide Pulse Modulation on both port 1 and port 2.
- This dialog may look different depending on the PNA model and number of receivers available.

Configures either the internal pulse generators (PNA-X models with relevant options), or Agilent 81110A Pulse Generator outputs. You can configure each 81110A Pulse Generator with either one or two 81111A output modules.

The Source Mod and four PNA receiver gates can each have their own, or shared, pulsed generators allowing identical **Width** and **Delay** values which are selected on the Main dialog.

To share an external generator output between one or more PNA inputs, use the same GPIB address and output module for each PNA input.

Internal Pulse Gen Output (available ONLY on the [PNA-X opt 025](#))

Specify the Pulse Gen (1 through 4) to use to modulate each of the PNA receiver IF gates or Sources.

External Pulse Generator settings

GPIB Addr: The GPIB address of the 81110A.

Output: The output module of the 81110A.

Master: The 81110A that uses the 10 MHz reference signal from the PNA.

Enabled: Turns the pulse output ON.

External Gate/Modulator settings

High: Specify a 'TTL-High' voltage level

Low: Specify a 'TTL-Low' voltage level

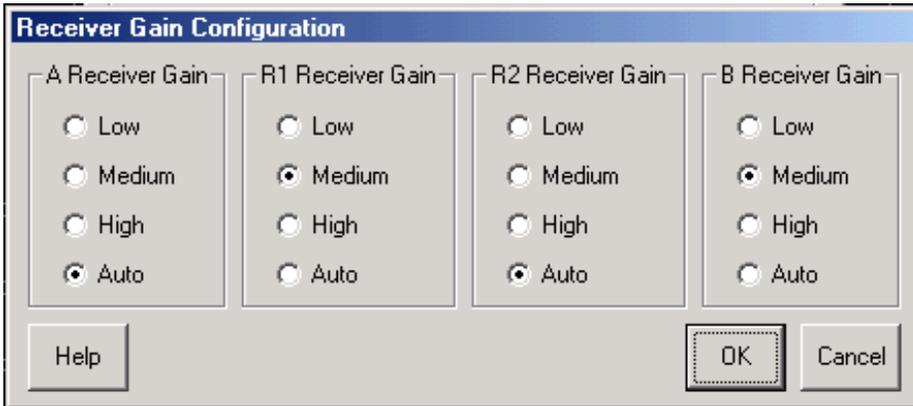
Ext Impedance: Impedance of the modulator used to create the pulse.

Complement: When this box is cleared, TTL HIGH is the pulse. When checked, TTL LOW is the pulse.

Using Internal Modulators When this box is checked, the voltage, impedance, and complement values are forced to settings that prevent damage to the internal modulator.

Using Internal Pulse Generators Makes the appropriate settings on this dialog available.

Using Internal PNA gates When this box is checked, the voltage, impedance, and complement values are forced to settings that prevent damage to the internal gates.



Receiver Gain Configuration dialog box help

[See Block Diagram of PNA-X IF Path / Pulse Generators / Source Modulation](#)

This dialog may look different depending on the PNA model and number of receivers available.

Sets the gain of each PNA receiver manually or automatically.

Auto - The PNA selects the best gain level to make pulsed measurements.

Use the following to manually set the gain for each receiver.

Low - about 0 dB of gain

Medium - about 17 dB of gain

High - about 24 dB of gain

The PNA-X has the following attenuation settings:

Low - 30 dB of attenuation

Medium - 15 dB of attenuation

Hi - 0 dB of attenuation

Calibration in Pulse Mode

To perform a calibration in pulse mode (option H08), first configure and apply the pulse parameters (PRF, Pulse Width, Delays, IF gating, and so forth) **before** calibrating the system. This will ensure the PNA is configured properly during the calibration and measurement.

When performing [Unknown Thru](#) or [TRL calibrations](#), ALL receivers must be gated. Otherwise, the error terms will not be correct after the calibration has completed. This can be accomplished by either having a separate pulse generator output for each of the IF gates, or by connecting pairs of the IF gates

together with BNC-T's. For example, if the pulse generator does not have enough outputs, then connect the R1 and R2 IF gates to the same pulse generator output. Also, connect the A and B IF gates to either separate outputs (recommended) or one output (reduces flexibility). The error terms will then be valid after the calibration is complete.

Pulse Profiling

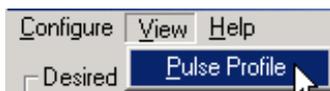
Pulse profiling provides a time domain view of the pulse envelope. Profiling is performed using a measurement technique that "walks" a narrow receiver "snapshot" across the width of the pulse. This is analogous to using a camera to take many small snapshots of a wide image, then piecing them together to form a single, panoramic view.

- Pulse Profiling can be performed using ratioed or unratioed measurements.
- Pulse Profiling is performed at a single CW frequency.

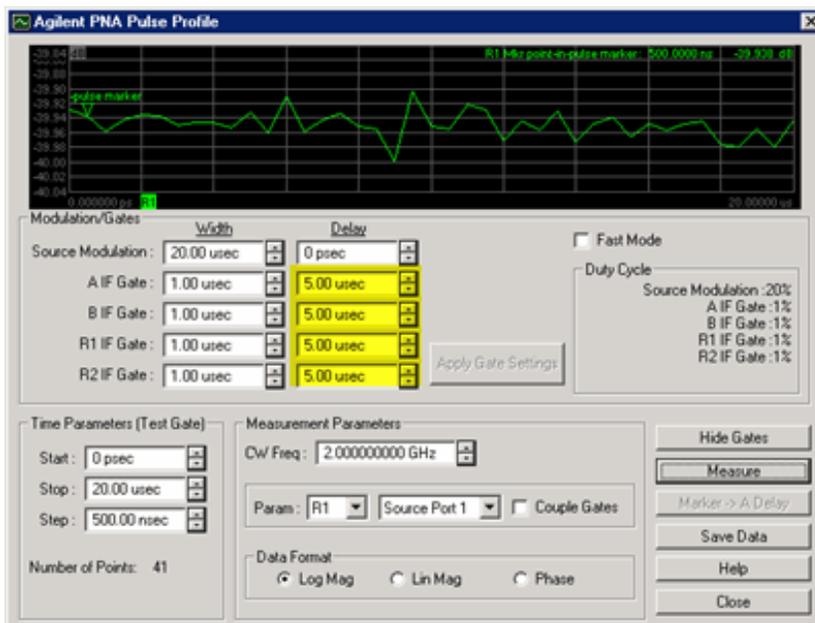
How to perform Pulse Profiling

From the [Pulse App main dialog box](#),

Click the **Pulse Profile** button. or:



If this setting is unavailable, check **Pulse Mode ON**, click **Calculate**, then **Apply**.



Pulse Profile dialog box help

[Learn about Pulse Profiling \(scroll up\)](#)

[See Block Diagram of PNA-X IF Path / Pulse Generators / Source Modulation](#)

Modulation / Gates

These settings duplicate those found on the main [Pulse App dialog box](#).

In Pulse Profile, the Gate Delay settings (highlighted in yellow) are significant only with certain **Measurement Parameter** and **Couple Gates** settings.

Time Parameters

Start, Stop These two combine to make the window of the assembled pulse profile. To view the entire pulse, the start and stop values must be at least as wide as the Source Modulation **Width** plus **Delay** value.

Step Each consecutive snapshot is incremented by this value until the stop value is reached. Therefore, the number of points for the pulse profile measurement can be calculated as: $(\text{Stop} - \text{Start}) / \text{Step}$. The higher the number of points, the longer it takes to make the measurement.

Measurement Parameter

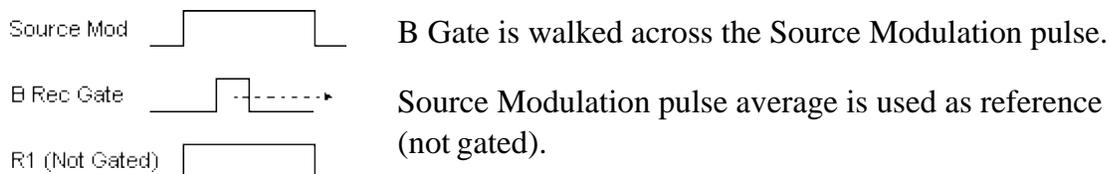
CW Freq. Frequency of the PNA source.

Source Port The PNA port supplying the source power. Only required for single receiver (unratioed) measurements.

Param(eter) Only those receiver gates (and relevant measurements) that are configured in [Pulsed Generator Configuration](#) are available.

Note: When a single receiver (unratioed) is selected, **Gate Delay** Settings (highlighted in yellow on above dialog image) are ignored.

If the reference receiver gate is NOT configured, the average of the Source Modulation pulse is used as the reference. **For example:** With **S21** Selected, but **ONLY B** receiver gate is configured, then...

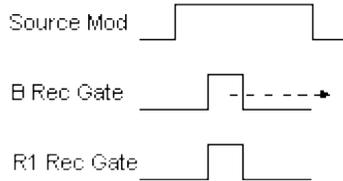


Coupled Gates Used when the appropriate receiver gates are [configured](#) for your **S-parameter** measurement ONLY. This setting is ignored when a single receiver (**Param**) is selected.

- **Uncoupled** (box cleared) The reference gate is **FIXED** at the delay setting as the test gate is walked across the Source Modulation pulse as dictated by the **Time Parameter** settings.

For example:

S21 Selected, B and R1 receiver gates configured, Gates Uncoupled



- **Coupled** (box checked) The reference gate is walked synchronously with the test gate as dictated by the **Time Parameter** settings. Only the **difference** between the test and reference gate delay values is significant; NOT the absolute values.

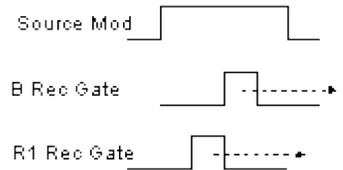
For example:

S21 Selected, B and R1 receiver gates configured, Gates Coupled

B gate delay = 3 microseconds,

R1 gate delay = 2 microseconds

Difference = 1 microsecond



B gate leads R1 gate by 1 microsecond.

Data Format Log Magnitude, Linear Magnitude, or Phase (only available if S-parameter selected).

Buttons

Show Gates Allows you do change the receiver gating width and delay while looking at the results.

Apply Gate Settings Click after making changes to gate settings.

Continuous Sweep Check, then click **Measure**, to continuously measure pulse profiling.

Measure Click to start the pulse profile measurement. Becomes **Stop** when continuously sweeping.

Marker to Delay After making a measurement, you can drag the display maker to any point along the trace. Click this button and the marker time is entered into the Receiver **Delay** field on the [main dialog box](#).

Save Data Saves time domain data to the PNA hard drive in any of the following formats:

- Touchstone (*.s1p)
- Comma delimited (*.prn)
- Citifile (*.cti)

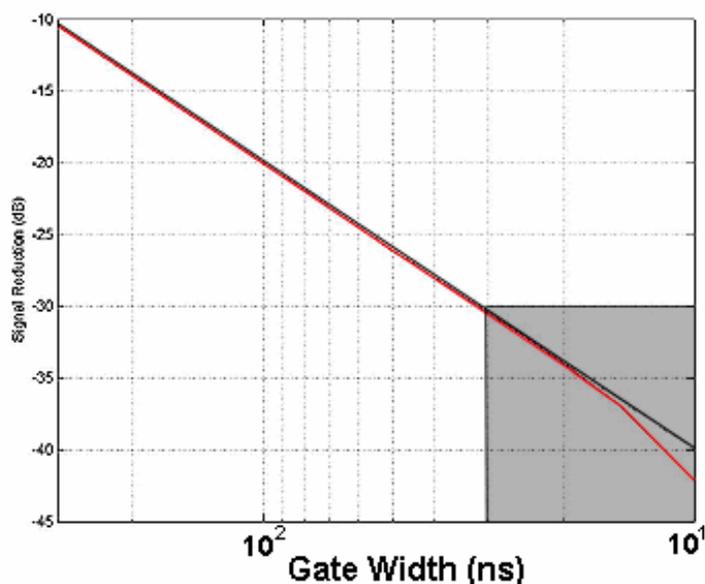
[Learn more about these data formats](#)

Signal Reduction versus Gate Width

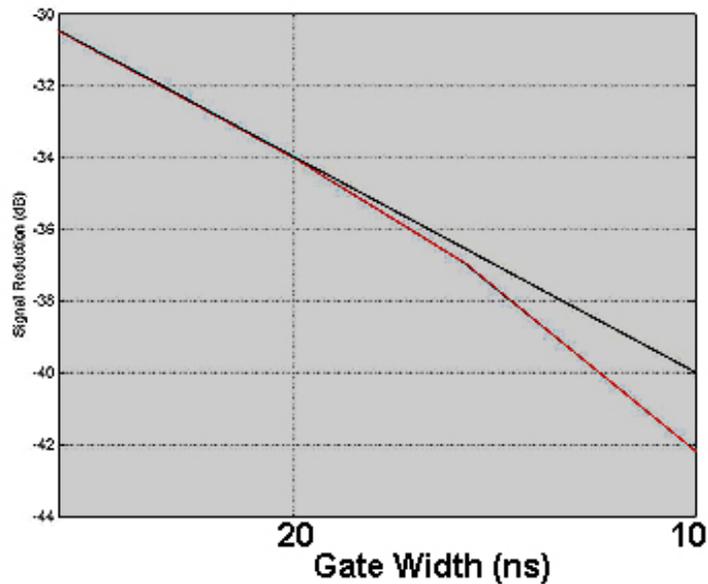
Signal Reduction versus Gate Width

PRF = 1 MHz

The following two figures show the performance of the internal IF gates as the width is narrowed.



The following is a zoomed image of the shaded area (above).



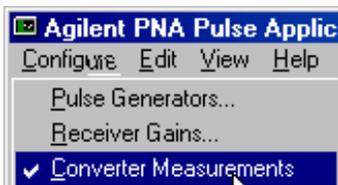
- The straight line shows the theoretical loss in dynamic range due to duty cycle effects when using narrowband detection.
- The curved (red) line shows the actual measured performance of the gates.
- The minimum gate width for <1dB deviation from theoretical is approximately 20ns.

See the specifications for the [option H11](#) and [option H08](#).

Pulsed Frequency Converter Measurements

The Narrowband Pulsed Application works with both [FCA \(option 083\)](#) and standard [Frequency Offset \(opt 080\)](#) measurements. On the **Configure** menu, check **Converter Measurements**. When checked, this setting prevents the Narrowband Pulsed Application from [overwriting frequency offset values](#). This may limit the number of **PRF** and **IFBW** solutions that are returned when **Calculate** is pressed on the main Pulsed Application dialog box.

Note: Pulse Profiling can NOT be performed with frequency converter measurements.



Writing your own Narrowband Pulsed Application

You can use the Narrowband Pulsed Application or use an example program as a template for making your own Narrowband Pulsed Application.

The Narrowband Pulsed Application uses a custom .dll to perform the calculations that are necessary to make pulsed measurements. Use the COM Method below to send and return values to **agilentpnapulsed.dll**. Then use SCPI or COM commands to control the PNA.

COM Example Program	PNA-X Create
SCPI Example Programs	Point-in-Pulse Pulse Profile
COM Methods	ConfigEnhancedNB2 ConfigEnhancedNBIFAtten
SCPI commands	SCPI
COM commands	COM

Install and Register the Pulsed .dll on your PC

To create your own Narrowband Pulsed Application, or run the Narrowband Pulsed Application from a remote PC, you must do the following:

1. Copy the following files from the PNA C:/program files/agilent/network analyzer/ to a directory on your PC.
 - **agilentpnapulsed.dll**
 - **OffsetList.txt**
 - **prfbw.txt**
 - **prfbwmixer.txt**
2. To register the ActiveX DLL in Microsoft Windows Operating System:
 - From a command prompt on your PC, navigate to the directory where you copied the DLL.
 - Type: **regsvr32 agilentpnapulsed.dll** and press **Enter**

For Operating Systems other than Windows, see their associated help files to learn how to register DLL files.

29-Apr-2009 Added Opt 036 and 037 note (A.08.60)
4-Sep-2008 Removed legacy content
2-Jul-2008 Added links to SCPI examples

Swept IMD and IM Spectrum Concepts

- [Swept IMD Concepts](#)
- [Swept IMD for Converters](#) (separate topic)
- [Swept IMD Parameters](#)
- [How the PNA Measures IMD](#)
- [How an IM Spectrum Channel Works](#)
- [IM Spectrum Parameters](#)

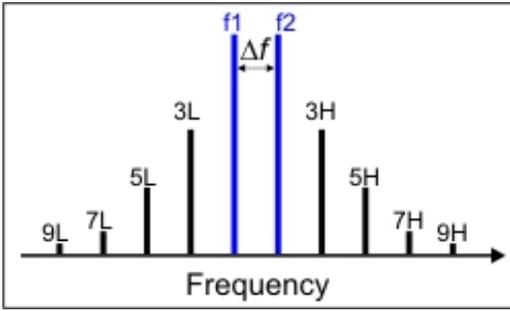
Other IMD topics

Swept IMD (Intermodulation Distortion) Concepts

When a device or system is subjected to multiple input frequencies, the non-linearity of the DUT can generate undesired outputs at other frequencies. Typically, two input tones of equal power separated in frequency by a specified amount are used to stimulate the device while observing the resulting frequency spectra at the output. A variety of measurements can then be utilized to determine the intermodulation distortion characteristics of the device.

The frequencies of the resulting distortion products are predictable. While many mixing products can be generated, the high and low signals of the "odd order" products (3rd, 5th, and so forth) are close enough to the original two signals to potentially interfere with adjacent communication channels. With the exception of the 2nd order product, the higher "even order" products are usually far enough away to be of no interest.

The following image and table shows two equal-power main-tones and the nearby odd-order distortion products, as well as the 2nd order product (not shown in the image). Notice that the frequency separation between adjacent odd-order products is the same as the separation of the main tones (Delta F) frequency. For most devices, these distortion products become worse as the device is pushed further into compression.



Two main tones (f_1 and f_2) with odd order intermodulation products.

The following table shows the calculations and example frequencies (**Blue text**) of the intermodulation products that are closest to the two main tones.

Product	Low (L)	High (H)
Main tones	f1	f2
	100 MHz	120 MHz
3	$2f_1 - f_2$	$2f_2 - f_1$
	80 MHz	140 MHz
5	$3f_1 - 2f_2$	$3f_2 - 2f_1$
	60 MHz	160 MHz
7	$4f_1 - 3f_2$	$4f_2 - 3f_1$
	40 MHz	180 MHz
9	$5f_1 - 4f_2$	$5f_2 - 4f_1$
	20 MHz	200 MHz
2	$f_2 - f_1$	$f_1 + f_2$
	20 MHz	220 MHz

[Learn about Swept IMDx for Converters Concepts](#)

Swept IMD Parameters

The following basic parameters, offered for both Amps and Converters, are expanded to over 150 by selecting specific product tones (2,3,5,7,9), the Low-side, High-side, or Average of these tones, measured at the Input or Output of the DUT.

- [Tone Power](#)
- [Tone Gain](#)
- [Intermodulation Distortion](#)
- [Intercept Point Parameters](#)
- [Composite Triple Beat \(CTB\)](#)
- [Composite Second-order Beat \(CSO\)](#)
- [Cross-Modulation Distortion](#)

[Learn how to select these IMD parameters.](#)

Tone Power Parameters

Tone Power parameters measure the **absolute** power level of the main tones, odd-order product tones up to the 9th order, and the 2nd order product tones. These tone powers can be measured at the input and output of the DUT. Because the tones come in pairs, the **Low tone**, **High tone**, and the **Average** of the two can be measured and displayed.

The Average Tone Power is calculated as follows:

$$\text{Avg} = (\text{High tone (dBm)} + \text{Low tone (dBm)}) / 2$$

When measuring the 2nd order products, only the Low tone and High tones are allowed. When the main tones are separated by less than 10 MHz, the Low tone (f2-f1) is below the frequency range of the PNA.

Tone Gain

Tone Gain (in dB) calculates the main tone Output Tone power / Input Tone power. Because the tones come in pairs, Tone Gain can be calculated for the **Low tone**, **High tone**, and the **Average** of the two as indicated in the Average Tone Power calculation.

For IMDx for Converters, the Input and the Output tones are typically at different frequencies.

Intermodulation Distortion Parameters

IMD parameters measure the **difference** in power level between the specified product tone and the main tones. These IMD parameters are calculated from the Tone Power measurements. IMD parameters can measure the odd-order product tones up to the 9th order, and the 2nd product tones, at the DUT Input or Output. For each specified product, the difference between the **Low** product and main tone, difference between the **High** product and main tone, and difference between the **Averages** of the product and main tone can be measured and displayed.

Swept IMD supports IMD parameters which are calculated as follows:

$$\begin{aligned} \text{IMxLo} &= \text{Pwr}\text{xLo} - \text{Pwr}\text{MainLo} \\ \text{IMxHi} &= \text{Pwr}\text{xHi} - \text{Pwr}\text{MainHi} \\ \text{IMx} &= \text{Pwr}\text{x} - \text{Pwr}\text{Main} \\ \text{IMxLoIn} &= \text{Pwr}\text{xLoIn} - \text{Pwr}\text{MainLoIn} \\ \text{IMxHiIn} &= \text{Pwr}\text{xHiIn} - \text{Pwr}\text{MainHiIn} \\ \text{IMxIn} &= \text{Pwr}\text{xIn} - \text{Pwr}\text{MainIn} \end{aligned}$$

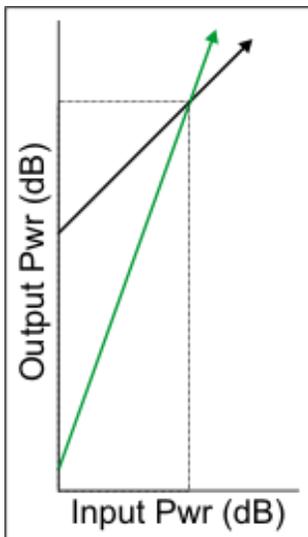
where:

- **x** = the IM product of interest (2, 3, 5, 7, 9)
- **Avg** is implied if **Hi** or **Lo** is not stated
- **Output** is implied when **In** is not stated

[Learn how to select IMD parameters.](#)

Intercept Point Parameters

As the main tone output power increases (black arrow), output power in the specified product tone increases at a predictable, and steeper, rate (green arrow). At some point, the power in the product tone will be equal to the power in the main tone. The power level at which this occurs is known as the intercept point. Measuring this point directly is typically not possible. Therefore, it is calculated by measuring the main tone power and the specified product tone power.



The Swept IMD App can display either the DUT Input power or DUT Output power that is required to

achieve the theoretical intercept point. This is called either Input Referred (IIP) or Output Referred (OIP).

This measurement can be made for the 2nd, 3rd, 5th, 7th, and 9th order intercept points. In addition, the measurements can be made for either the Low tone, the High tone, or the Average of the two. However, for the 2nd order intercept point, only Low and High tone parameters are supported; not Average.

Swept IMD supports Intercept Point parameters which are calculated as follows:

$$\begin{aligned} \text{OIPxHi} &= \text{PwrMain} - \text{IMxHi}/(\text{x}-1) \\ \text{OIPxHiIn} &= \text{PwrMain} - \text{IMxHiIn}/(\text{x}-1) \\ \text{OIPxLo} &= \text{PwrMain} - \text{IMxLo}/(\text{x}-1) \\ \text{OIPxLoIn} &= \text{PwrMain} - \text{IMxLoIn}/(\text{x}-1) \\ \text{OIPx} &= \text{PwrMain} - \text{IMx}/(\text{x}-1) \\ \text{OIPxIn} &= \text{PwrMain} - \text{IMxIn}/(\text{x}-1) \\ \text{IIPxHi} &= \text{PwrMainIn} - \text{IMxHi}/(\text{x}-1) \\ \text{IIPxHiIn} &= \text{PwrMainIn} - \text{IMxHiIn}/(\text{x}-1) \\ \text{IIPxLo} &= \text{PwrMainIn} - \text{IMxLo}/(\text{x}-1) \\ \text{IIPxLoIn} &= \text{PwrMainIn} - \text{IMxLoIn}/(\text{x}-1) \\ \text{IIPx} &= \text{PwrMainIn} - \text{IMx}/(\text{x}-1) \\ \text{IIPxIn} &= \text{PwrMainIn} - \text{IMxIn}/(\text{x}-1) \end{aligned}$$

where:

- **x** = the IM product of interest (2, 3, 5, 7, 9)
- **PwrMain** = average power of the main tones at the DUT Input or Output
- **IMx** = main tone power - product tone power (from [above IMD parameter](#))
- **Avg** is implied if **Hi** or **Lo** is not stated
- **Output** is implied if **In** is not stated

[Learn how to select IMD parameters.](#)

Composite Triple Beat (CTB)

From the NCTA Standard, composite triple beat is defined as the modulation beat of the target channel signal caused by triple beat resulting from the nonlinear characteristic of the DUT. Composite triple

beat is expressed as the ratio of the target channel signal level to the maximum mean level of beat components dispersed around the carrier of that target channel.

Swept IMD supports two parameters of this type:

- CTB is based upon an approximation for the number of beats in mid-band
- CTBE is based upon an approximation for the number of beats at the band edge.

The equations for these two parameters are as follows:

$$\text{Mid-Band CTB (dB)} = -2(P_i - P_s) + 6 + 10\log(3N^2/8) + \text{CTB Offset}$$

$$\text{Band Edge CTBE (dB)} = -2(P_i - P_s) + 6 + 10\log(N^2/4) + \text{CTB Offset}$$

Where:

- P_i = Output power level at the third order intercept point (dBm): OIP3 (Lo | Hi)
- P_s = One of the following values based upon the Composite Normalization Mode:
 - For **PDBM** or **PDBMV** mode, P_s = CompositeNormalizedCTBPower
 - For **Number of Carriers** mode, P_s = PwrMain (AVG) – 10Log(N/2)
 - For **None** mode, P_s = PwrMain (AVG)
- **CTB OFFset** = Offset value for CTB calculation
- **N** = Total number of carriers.

Note: **CTB OFFset** and **N** values can ONLY be set using [SCPI or COM commands](#).

[Learn how to select IMD parameters.](#)

Composite Second-Order (CSO)

From the NCTA Standard, composite second order is defined as the modulation beat of the target channel signal caused by second order beat resulting from the nonlinear characteristic of the DUT. Composite second order beat is expressed as the ratio of the target channel signal level to the maximum mean level of beat components dispersed around 0.75 MHz and 1.25 MHz above and below the carrier of that channel.

Swept IMD supports a CSO parameter which is calculated as follows:

$$\text{CSO (dB)} = (P_i - P_s) + 10\log(N) + \text{CSO Offset}$$

Where:

- P_i = Output power level at 2nd order intercept point: OIP2 (Lo | Hi)
- P_s = One of the following values based upon the **Composite Normalization Mode**:
 - For **PdBm** or **PdBmV** mode, $P_s = \text{CompositeNormalizedCSOPower}$
 - For **Number of Carriers** mode, $P_s = \text{PwrMain (AVG)} - 10\text{Log}(N/2)$
 - For **None** mode, $P_s = \text{PwrMain (AVG)}$
- **CSO OFFset** = Offset value for CSO calculation
- **N** = Number of distortion products.

Note: **CSO OFFset** and **N** values can ONLY be set using [SCPI or COM commands](#).

[Learn how to select IMD parameters.](#)

Cross-Modulation Distortion

From the NCTA Standard, cross modulation is defined as the distortion that causes modulated carrier components of undesired channels to amplitude-modulate the target channel carrier due to the nonlinear characteristic of the unit under test. Cross modulation distortion is expressed as the ratio of the target channel carrier level to the level of modulated components of the carrier of the target channel resulting from modulated signals of undesired channels.

Swept IMD supports an XMOD parameter which is calculated as follows:

$$\text{XMOD} = -2(P_i - P_s) + 6\text{dB} + 20\text{Log}(N)$$

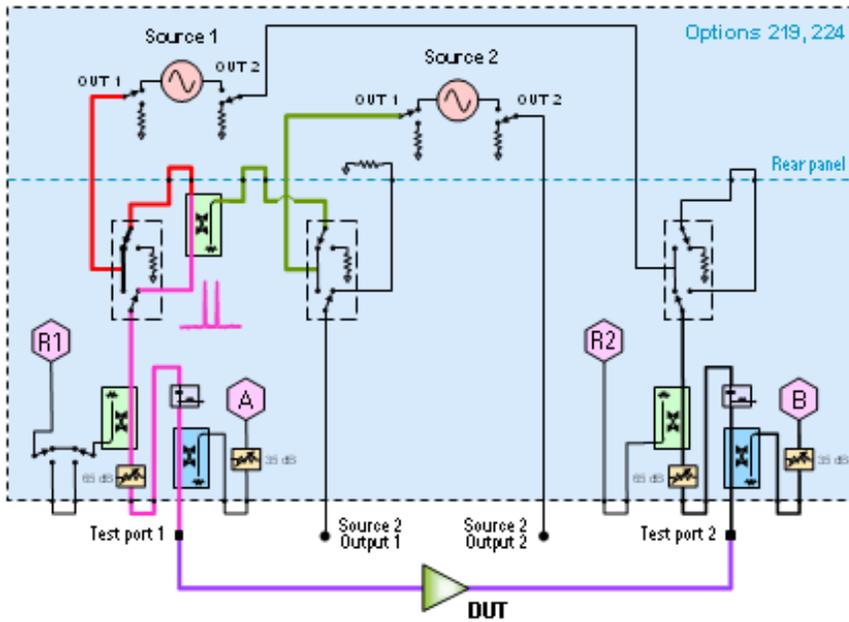
Where:

- P_i = Output power level at third order intercept point: OIP3 (Lo | Hi)
- P_s = Power level of each carrier: PwrMain (AVG)
- **N** = Total number of carriers.

Make Cross Modulation settings using [SCPI or COM commands](#).

How the IMD Application Works

The following diagram illustrates how the PNA is configured to generate the two main tones. This shows a PNA-X with dual sources and the internal combiner. A 2-port or 4-port N522xA model can also be used. Learn how to [Configure External Source and Combiner](#).



2-port PNA-X generates the f_1 and f_2 main tones.

Depending on the specified parameters and sweep type, the sources and receivers are tuned to the appropriate frequencies in order to measure all of the required main and product tone powers. For example, an IM3 parameter requires the measurement of both main tones, and the 3rd order High and Low tone powers.

The [Narrowband IF path](#) is used for IMD measurements to help reduce spurious responses. Because the narrowband filter has a bandwidth of about 28 kHz, using an IFBW greater than 30 kHz does nothing to improve measurement accuracy. [Learn how to set IFBW for IMD.](#)

Limiting Stimulus Settings and Out of Range Product Tones

Because the main tones are generated by the PNA internal sources and external sources, the frequencies of the **main tones** must always be within the frequency range of the PNA or external source. Sweep parameter values are adjusted when necessary to ensure that f_1 and f_2 frequencies are within these limits.

However, the PNA DOES allow you to make settings that cause the selected **IM products** to fall outside the frequency range of the PNA. For example, with the main tones at 10 MHz and 15 MHz, the PNA will allow you to select the parameter **IM3Lo** (3rd low side product tone). However, the frequency of this product will be at $2f_1 - f_2$ or 5 MHz, which is below the frequency range of the PNA. In these cases, the trace data is set to zero, which converts to -200 dB in Log Mag format.

Limited Number of Acquisitions

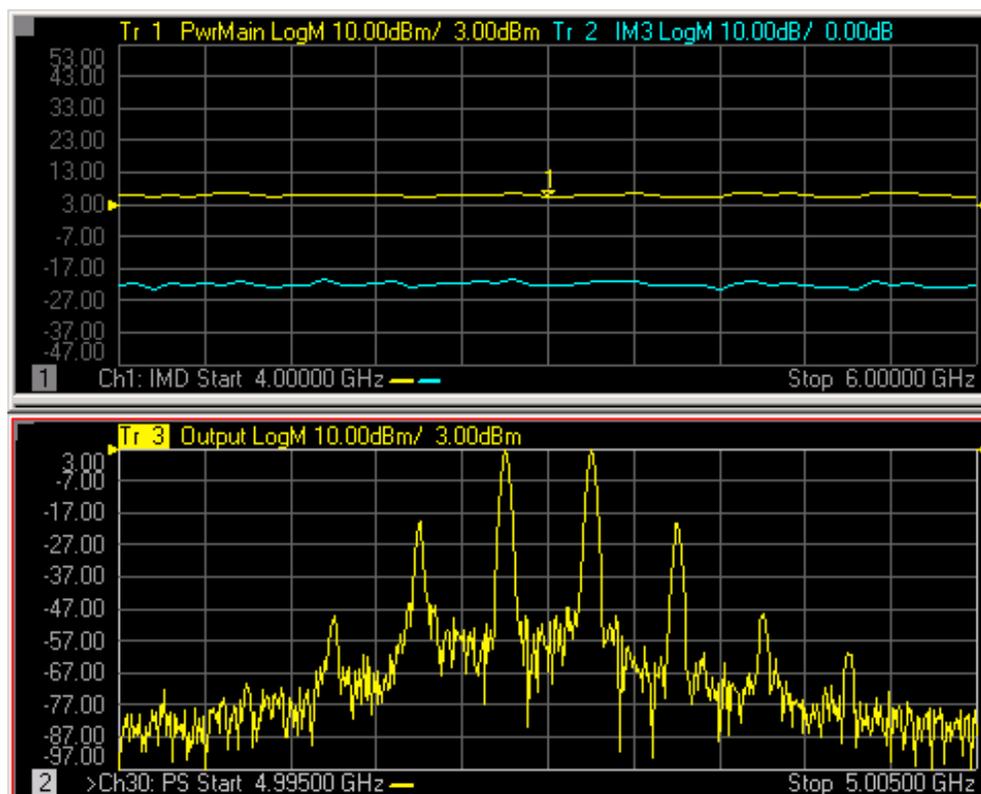
The total number of acquisitions per sweep can not exceed 32,001 points. The number of acquisitions

is determined by multiplying the number of trace points, by the number of tones frequencies, then by 2 (for both Input and Output frequencies). The PNA will automatically reduce the number of trace points to ensure the total number of acquisition points does not exceed 32,001.

How an IM Spectrum Channel Works

Before reading this topic, you should become familiar with [IMD Concepts](#).

The IM Spectrum channel provides a traditional spectrum analyzer view of the intermodulation distortion behavior of a device. Unlike the Swept IMD channel, the main tones (F1 and F2) are fixed while the receiver is swept over a frequency range of interest in order to generate a display as shown below.



IM Spectrum trace (bottom) with Swept IMD traces (top)

A typical spectrum analyzer does NOT have a signal source. This one does. The signal source, or stimulus, settings for the F1 and F2 main tones can be set in either the IM Spectrum channel or the Swept IMD channel. These settings include the frequencies and power levels of the main tones.

Receiver Settings

The settings for the IM Spectrum receiver can be set ONLY in the IM Spectrum channel. These settings include how many tone products to view - which determines the center and frequency span - and the

Resolution Bandwidth.

You can choose from several Resolution Bandwidths which also determines the number of data points used in the channel. The higher the Res BW, the fewer the number of data points. The formula for determining the number of points is:

$$N = (\text{Span}/\text{ResBW}) * 3$$

The IM Spectrum channel performs multiple measurements for each data point in order to reject unwanted images which are generated by the PNA internally. This provides a high degree of confidence that signals captured in a trace are real and are not spurious responses generated in the measurement process.

IM Spectrum Parameters

You can select from three different IM Spectrum parameters.

1. The tones OUT of the DUT (default parameter).
2. The tones IN to the DUT (to be sure that the input signals are pure). **NOT supported** in IMx Spectrum
3. Reflected tones off the DUT input. **NOT supported** in IMx Spectrum

Learn how to select IM Spectrum parameters for [Amplifiers](#) or [Converters](#).

Learn all about IM Spectrum for [Amplifiers](#) or [Converters](#).

[See list of all IMD topics.](#)

Last Modified:

9-Apr-2013	Fixed typo
12-Mar-2009	Added Tone Gain (8.5)
9-Mar-2009	MX New topic

Swept IMD for Amplifiers (Opt 087)

- [Features and Limitations](#)
- [Create a Swept IMD Measurement](#)
 - [Frequency tab](#)
 - [Power tab](#)
 - [Configure tab](#)
 - [How to specify IMD Parameters](#)
- [IMD Calibration](#)
- [Saving Swept IMD Power Data](#)

Other IMD (Opt 087) Topics

Features and Limitations

[See requirements.](#)

Features

- Fast and easy setup for the measurement of a variety of distortion related parameters up to 9th order.
- Measurement at both input and output of a DUT.
- Supports a variety of sweep-modes for the main-tones: center-frequency linear and segment sweep, tone-separation sweep, power sweep, or CW sweep (fixed main-tone).
- Make very fast, accurate measurements using the PNA sources with high-power, high linearity, and low harmonics.
- Supports calibration and correction of Swept IMD parameters.
- Independently set IFBW for measuring main-tones versus product tones.

Limitations

- 2-port non-frequency converters ONLY. For frequency converters, use [Swept IMDx](#).
- DUT to PNA port mapping is limited. Port selections are made on the Power tab. [Learn more.](#)

- When using [Integrated Pulse application](#), the IF Filter setting for the relevant receiver must be changed to 'Wide'. [Learn how](#). The default IF Filter setting in Swept IMD is 'Narrow' in order to avoid spurs and harmonics. The IF Attenuator setting (on the same dialog) may also require adjusting.

The following features are NOT available with Swept IMD:

- [Number of points limited to 20,001](#)
- Independent IFBW, Power Levels, or Sweep Time in a [segment table](#) is NOT supported.
- Analog Sweep ([Stepped sweep](#) mode only)
- [Log frequency](#) sweeps
- [Unratioed receiver measurements](#) (A, B, R)
- [ECal User Characterization](#)
- [Time Domain](#)
- [Balanced measurements](#)
- Save [Formatted Citifile](#) data.
- [Save SnP data](#).
- [External sources](#)
- [Interface Control](#)
- [Port extensions](#)
- [Some Fixturing Features](#)
- [External Test Set Control](#) (Option 551)
- [Integrated Narrowband](#) or [Narrowband Pulse App](#)
- [External DC Sources](#) (DC Meters ARE supported).
- [See Frequency limitations in a Swept IMD channel](#).

Create a Swept IMD Measurement

1. On the PNA front panel, press **Meas** then **[Measurement Class]**
2. Select **Swept IMD**, then either:
 - **OK** delete the existing measurement, or
 - **New Channel** to create the measurement in a new channel.

3. A Swept IMD measurement is displayed. To select additional parameters to display, click **Response**, then **Measure**, then select a parameter from the list.

How to start the Swept IMD Setup dialog

To provide quicker access, use the Setup softkey. [Learn how.](#)

Using front-panel HARDKEY [softkey] buttons

1. Press **FREQ**
2. then **[Swept IMD Setup]**

Using a mouse with PNA Menus

1. Click **Stimulus**
2. then **Frequency**
3. then **Swept IMD Setup**

Programming Commands

Frequency tab - IMD Setup dialog box help

Sweep Type

- Sweep fc
- Sweep DeltaF
- Power Sweep
- CW
- Segment Sweep fc

Sweep Settings

Start fc: 10.500000 MHz

Stop fc: 1.00000000 GHz

Center fc: 505.250000 MHz

Span fc: 989.500000 MHz

Fixed DeltaF: 1.000000 MHz

Number Of Points: 3

Main Tone IFBW: 1.000 kHz

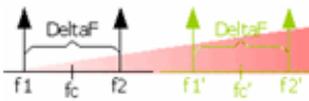
IM Tone IFBW: 1.000 kHz

Reduce IF BW at Low Frequencies

Configures the Sweep Type and frequency range for SweptIMD and [Swept IMDX](#) measurements.

Sweep Type and Sweep Settings

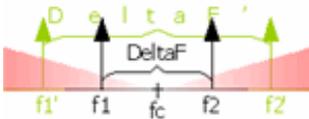
Sweep f_c (center frequency)



Maintaining a constant tone spacing (Fixed ΔF) and tone power, f_c is swept from Start f_c to Stop f_c . Center Frequency can also be specified as Center f_c and Span f_c .

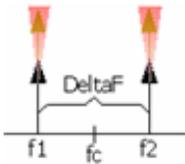
At each f_c , the receivers are tuned to all of the required frequencies to measure the power of the appropriate tones.

Swept ΔF (tone spacing)



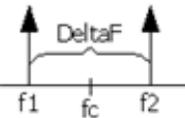
The specified f_c (center frequency) and tone power is held constant. The tone spacing is increased from Start ΔF to Stop ΔF in the specified number of points.

Power Sweep



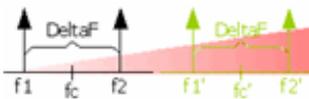
The main tone frequencies are specified as either f_1 and f_2 , or as f_c and ΔF . These frequencies are held constant while the power of each main tone is varied from the Start-Power to Stop-Power in the specified number of power points. The power of each tone can be set (on the Power tab) individually or as a pair by checking Coupled Tone Power.

CW



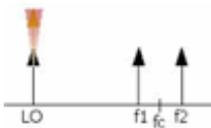
The main tone frequencies and power levels are held constant. Measurements are taken for the specified Number of Points. The X-axis is number of points.

Segment Sweep f_c (Swept IMD ONLY)



Same as Sweep f_c except that the center frequencies are constructed using the standard segment table. [Learn how.](#)

LO Power Sweep (Swept IMDX ONLY)



The main tone frequencies and power levels are held constant. Measurements are taken for the specified number of points. The X-axis is LO Power.

Segment Sweep Notes: (Swept IMD ONLY)

- The segment table shown on the dialog is 'READ-ONLY'.
- Learn how to [Create and edit the Segment Sweep table.](#)

- **Independent IFBW** and **Power** are NOT available.
- [X-axis point spacing](#) is available beginning with A.09.10.

Number of Points Enter the number of data points for each sweep. See Limited Number of Acquisitions.

IFBW

The [Narrowband IF path](#) is used for IMD measurements to help reduce spurious responses. Because the narrowband filter has a bandwidth of about 28 kHz, using an IFBW greater than 30 kHz does nothing to improve measurement accuracy.

Main Tone and IM Tone IFBW IF Bandwidth is specified separately for the main tones (f1 and f2) and for the intermodulation tones. This allows the higher-power main tones to be accurately measured at a higher - and faster - IFBW, while the lower-power product tones to be accurately measured a lower - and slower - IFBW.

Reduce IF BW at Low Frequencies - On the PNA, the trace noise becomes worse below 748 MHz. This is especially obvious between 10 MHz and 45 MHz. When this box is checked, the PNA uses a smaller IF Bandwidth than the selected value at frequencies below 748 MHz. [Learn more about the selected values.](#)

Power tab - IMD Setup dialog box help

Configures RF power and Power Sweep settings for IMD measurements.

Power ON (All channels) Check to turn RF Power ON or clear to turn power OFF for all channels.

DUT Input Port

Input Port Choose Port 1 or Port 3. When using Port 3, an external combiner is required. [Learn more.](#)

Source Attenuator Specifies the port 1 attenuator. This attenuator affects the range of available power into the DUT [Learn more about Source Attenuation.](#)

Receiver Attenuator This attenuation setting protects the A receiver from damage.

DUT Output Port

Output Port Choose Port 2 or Port 4 (with limitations), [Learn more.](#)

Source Attenuator This setting is used to improve the load match at the DUT output. Select 0 dB for power levels up to 10 dBm, and increase by 10 dB for every 10 dBm more output power.

Receiver Attenuator Specifies the attenuator setting for port 2. When the power into the receiver test port is around +10 dBm, the PNA receiver may be in compression. However, with receiver attenuation, lower input power levels may become too noisy to make accurate power measurements. In this case, lower IFBW for the IM tones to reduce noise. [Learn more about Receiver Attenuation.](#)

Tone Powers

Coupled Tone Power Check to set the same power level for each main tone using the f1 Power setting. Clear to set different f1 and f2 power levels.

ALC On Check to use internal ALC hardware (default). Clear to use Open Loop hardware. Open Loop leveling should only be used when doing [Wideband Pulse measurements.](#)

Power Leveling

Because the gain of the DUT can be different for the f1 and f2 tone frequencies, you can set tone power at either the input of the DUT OR the output using the following methods. Receiver Leveling will cause slower sweeps.

Set Input Power (Default) The specified f1 and f2 power levels are set at the DUT input. Input power level accuracy is based ONLY on the source power cal that is performed during the IMD cal. The input and output tones may NOT be equal or flat.

Set Input Power, receiver leveling The specified f1 and f2 power levels are set at the DUT input using receiver leveling at the input reference receiver. This ensures the tone power levels are equal at the DUT input. However, the output tones may NOT be flat due to variations in the gain of the DUT at different frequencies.

Set Input Power, equal tones at output The specified f1 and f2 power levels are set at the DUT

input and a measurement is made at the output. The inputs are adjusted once at each frequency to ensure the tone power levels are equal at the DUT output. However, the output tones may NOT be flat due to variations in the gain of the DUT at different frequencies.

Set Output Power, receiver leveling The specified f1 and f2 power levels are set at the DUT output. Receiver Leveling at the output receiver is used to accurately set the specified power level of each tone within the tolerance value that is set in the [Receiver Leveling dialog](#). This setting results in the output tones being equal and flat across the frequency range.

f1 / f2 Power

Fixed f1 Power Specify the power level for f1 at either the DUT input or output depending on the Power Leveling setting. Choose a value between -30 dBm and +30 dBm. When "Coupled Tone Power" is checked, power is set for both f1 and f2 tones.

Fixed f2 Power Available when Coupled Tone Power is NOT checked. Specify the power level for f2 at either the DUT input or output depending on the Power Leveling setting. Choose a value between -30 dBm and +30 dBm.

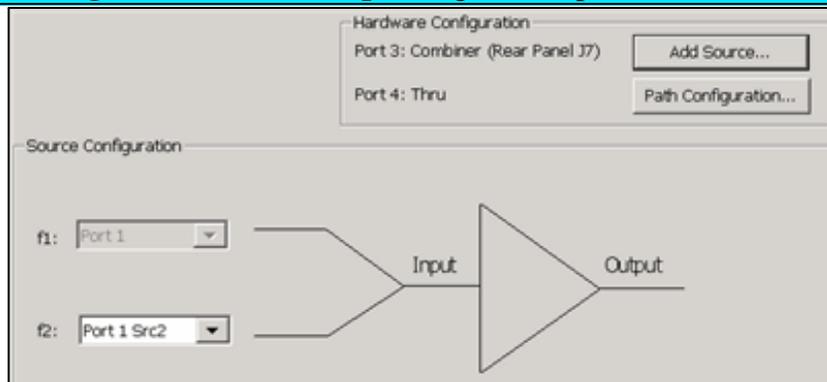
Start, Stop, and Step f1 and f2 Power Available when Power Sweep is selected on the [Frequency tab](#). Sets the Start and Stop power levels for f1 and f2, either individually or together with Coupled Tone Power checked.

Path Configuration click to launch the [RF Path Configuration](#) dialog.

Highlighted Note: RF2 tone power offset to compensate for combiner loss.

This message appears when the f2 tone is being supplied by an external source. The tone power has been increased on the external source to compensate for loss through the internal combiner. For example, if the tone power at the DUT should be 0 dBm, the power out of the source will be about 15 dB higher.

Configure tab- IMD Setup dialog box help



To accommodate single-source PNA models, an external source can be used for the f2 tone.

Learn how to [configure an external source and combiner](#) to make Swept IMD and IMDx measurements.

f1 Always uses PNA internal source 1.

f2 Select a source to be used for the f2 tone. This selection is available when an external source is configured and the **Active** box is checked on the [External Source Configuration dialog](#).

Buttons

Add Source Click to configure an external source using the [External Source Configuration dialog](#).

Path Configuration Click to launch the [Path Configuration dialog](#) (PNA-X models only).

How to add IMD Parameters

Using front-panel HARDKEY [softkey] buttons

1. Press **TRACES**
2. then [**New Trace**]
3. then select a parameter

Using a mouse with PNA Menus

1. Click **Trace/Chan**
2. then **New Trace**
3. then select a parameter

[Programming Commands](#)

IMD New Trace dialog box help

Param. Name	Type	Tone Select	Order	Measure At
<input type="checkbox"/> PwrMain	Tone Power	Avg	1	DUT OUT
<input type="checkbox"/> IM3	IMD Relative to Carrier	Avg	3	DUT OUT
<input type="checkbox"/> OIP3	Output Referred Intcpt. Pt.	Avg	3	DUT OUT
<input type="checkbox"/> IIP3	Input Referred Intcpt. Pt.	Avg	3	DUT OUT
<input type="checkbox"/> Pwr2	Tone Power	Avg	2	DUT OUT

Create In New Window
 Auto-Create Windows

Use this dialog to select IMD and [IMDx](#) parameters to measure and display.

Up to five parameters can be selected at a time, then click **Apply** to create those traces. Then select more without closing the dialog. There is no limit to the [number of traces and windows](#) allowed in the PNA.

Note: Calculations are NOT performed to determine if the frequency of a selected intermod (Order) product will be within the frequency range of the PNA. Measurements that fall outside of the frequency range of the PNA are displayed as -200 dB.

Param Name This name is built from the selected **Type, Tone Select, Order, and Measure At** settings. Once built, check to measure this parameter.

Type

Select the type of Swept IMD measurement to make.

The characters below (in parenthesis) are used in the Param Name.

- **Tone Power (Pwr)** Measures the absolute power of the specified (ordered) tone. [Learn more.](#)
- **IMD Relative to Carrier (IM)** Measures the power of the specified (ordered) tone relative to the power of the f1 or f2 tone. [Learn more.](#)
- **Input Referred Intercept Point (IIP)** From DUT measurements, calculates the theoretical power level at which the specified (ordered) intermod product will be the same power level as the carrier at the output of the DUT. The input power to the DUT at which this output power occurs is reported and displayed. [Learn](#)

[more.](#)

- **Output Referred Intercept Point (OIP)** From DUT measurements, calculates the theoretical power level at which the specified (ordered) product will be the same power level as the main tone at the output of the DUT. This value is reported and displayed. [Learn more.](#)
- **CTB Band-Edge Distortion (CTBE)** Composite 'Triple Beat' Distortion - 3rd order DUT OUT only. [Learn more.](#)
- **CTB Mid-Band Distortion (CTB)** Composite 'Triple Beat' Distortion - 3rd order DUT OUT only. [Learn more.](#)
- **CSO Distortion (CSO)** 2nd order DUT OUT only. [Learn more.](#)
- **XMOD 3rd Order Crossmod (XMOD)** 3rd order DUT OUT only. [Learn more.](#)
- **Tone Gain (ToneGain)** From tone power measurements, calculates the Output Tone power / Input Tone power for the specified tones. [Learn more.](#) The Input and Output tones are different frequencies.

Tone Select

Select the tone (High, Low, or Both) to be measured and displayed ([See image](#)).

- **High** Measure and display the power of the specified (ordered) tone on the High side of the main tones.
- **Low** Measure and display the power of the specified (ordered) tone on the Low side of the main tones.
- **Avg** Measure the specified (ordered) tones on both the High and Low sides. Then calculate and display the Average power level. **Note:** The **Avg** power level is actually the mean dB value of the high and low side tones - NOT the true average power level which would be calculated from linear power (watts). This is done to be consistent with the [Power Leveling algorithm](#) that is used to set the tone powers equal at the DUT input or output, which is also based on the dB values.
- **Max** Measure the specified (ordered) tones on both the High and Low sides. Then find and display the Maximum power level.
- **Min** Measure the specified (ordered) tones on both the High and Low sides. Then find and display the Minimum power level.

Used to build the parameter name:

- **Hi, Lo, Max,** and **Min** are appended to the Param Name when selected.
- Nothing is appended to the Param Name when **Avg** (default setting) is selected.

Important Notes - 2nd-order products

- 2nd-order products are likely to be outside of the PNA frequency range. Trace data will show all will show **all zeros** (linear) or **-200 dB** (log magnitude).
- If either the High or Low side falls outside the frequency range of the PNA, then **Avg** is NOT allowed.
- When displaying 2nd-order traces, **Avg** is NOT allowed, even when both the Lo and Hi products are displaying valid data. This is because 2 Low and 2 High products are usually very different from one another.
- When performing a calibration that is meant to include 2nd-order products, be sure [Include 2nd Order Products](#) is checked in the first Calibration dialog box.

Order

Specify the intermodulation product to measure. Choose from 1, 2, 3, 5, 7, 9.

- **Main** is appended to the Param Name when 1 is selected.
- Otherwise, the tone number is appended to the parameter name.

Measure At

Measure the selected parameter at either:

DUT Input

- **In** is appended to the Param Name (ex: PwrMainIn).
- The **input port reference receiver** is used to measure the fundamental tones and the required products.

DUT Output

- Nothing is appended to the Param Name (ex: PwrMain).
- The **output port measurement receiver** is used to measure the fundamental tones and the required products.

IMD Calibration

Overview

1. At the first page of the IMD Cal Wizard, you tell the calibration routine the frequencies at which the calibration is to be performed. Optionally, you can choose to perform the source power cal at only the center frequency midway between the main tones. For IMDx ONLY, you can also choose to perform a source power cal of the LO source.
2. The PNA calculates an array of source and receiver frequencies that incorporate all the main tone frequencies (low and high) and the specified product tone frequencies.
3. Using a power meter at port 1, a source and receiver calibration is performed to calibrate the R1 reference receiver to be a fast and 'tunable' power meter.
4. The R1 reference receiver is then used to perform a source power cal of the main tone frequencies: first the Source 1 / Low tone then the Source 2 / High tone. Both sources are left ON while each tone is measured in order to duplicate the impedance match under which the measurement will be performed.
5. Then a standard 2-port SOLT cal is performed at all frequencies using either an ECal module or mechanical standards. The 2-port cal is used to correct the source calibration R1 tracking terms for the match of the power sensor. It is also used to transfer the R1 tracking term to the B receiver.

Notes

- If the main tone frequencies change but are within the frequency range in which the calibration was performed, then the calibration becomes interpolated C*. This can occur by changing the start/stop/center/span frequencies, the number of points, or the sweep type. Learn more about [Interpolation](#).
- A Preference setting can be made which allows a Swept IMD and IMDx calibration to exceed the stop frequency limit of an ECal module. [Learn more](#).
- [Receiver calibrations](#) that are performed in a standard channel can be applied to a Swept IMD channel. However, [Source Calibrations](#) can NOT be applied.

See Also

[IM Spectrum Calibration](#)

How to start a Swept IMD Calibration

Using front-panel HARDKEY [softkey] buttons

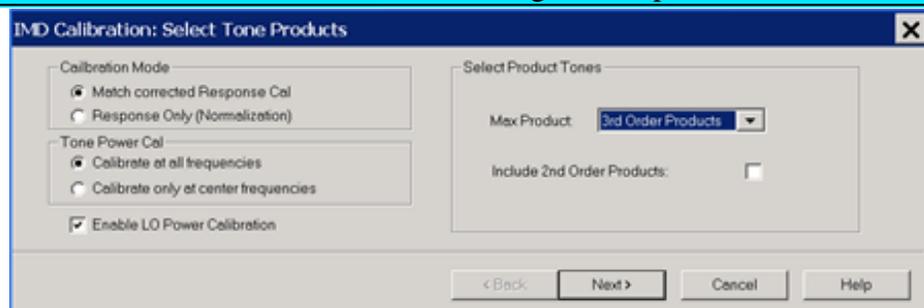
1. Press CAL
2. then [Start Cal]
3. then [Cal Wizard]

Using a mouse with PNA Menus

1. Click **Response**
2. then **Cal Wizard**

[Programming Commands](#)

Select Tone Products - IMD Cal dialog box help



Calibration Mode

Match corrected Response Cal This selection performs a full IMD calibration as described in the above [Calibration 'Overview'](#).

Response ONLY (Normalization) This IMD Cal does NOT correct for the mismatch of the power sensor. Choose this if you have a test configuration that does not easily accommodate making match measurements. Instead of a standard 2 port SOLT cal ([Step 5 above](#)), only the transmission tracking term is measured and used to transfer the R1 receiver tracking term (produced by the power sweep) to the B receiver.

Note: For the Response (Normalization) Cal, it is assumed that a [zero-length THRU standard](#) is being used to connect port 1 to port 2. If an adapter is used, there is NO compensation for delay or loss of the adapter. This can NOT be changed.

Tone Power Cal

Calibrate only at center frequencies The source power cal portion is performed at only the center frequency, which is midway between the main tones. This cuts the source power calibration time (the slowest part of the calibration) in half. The measurement is interpolated although the C* annotation is not shown in the status bar.

Calibrate at all frequencies The source power cal portion is performed at all main tone frequencies.

Enable LO Power Calibration (IMD_x ONLY) Check to perform a standard power calibration of the LO source as part of the calibration process.

Select Product Tones

Max Product Select the highest product that you will be measuring. The low and high frequencies for that product, and all lower 'odd' order products will be calibrated. For example, when 5th Order Product is selected, the frequencies for the Main Tones, and the Low and High

order products for the 3rd and 5th order products will be included in the calibration.

Include 2nd Order Products Check to calibrate the 2nd-order products in the frequencies to be calibrated. The frequencies of these products are usually far from the main tones.

Extrapolated ECal Calibration

A [Preference setting](#) can be made which allows a Swept IMD and IMDx calibration to exceed the stop frequency limit of an ECal module. The error terms beyond the ECal stop frequency are extrapolated.

The setting is designed to provide correction for order products (especially 2nd order) when the frequencies exceed the ECal module stop frequency.

- The accuracy of the extrapolated error terms is not as good as those that were measured.
- This feature is NOT allowed when using [Cal All Channels](#).
- The ECal Extrapolation state and frequencies are saved to the Cal Set and visible in the [Cal Set Properties](#) dialog.

Select DUT Connectors and Cal Kits - IMD Cal dialog box help

IMD Calibration: Select DUT Connectors and Cal Kits

DUT Connectors		Cal Kits	
Port 1	APC 3.5 male	Port 1	N4691-60003 User 1 ECal 00589
Port 2	APC 3.5 male	Port 2	N4691-60003 User 1 ECal 00589
Power Sensor	Ignored		

Modify Cal
OPTIONAL. Select [Modify Cal] to change the Cal Method and/or standards used for the selected cal kits.

View/Modify Source Cal Settings

< Back Next > Cancel Help

If **Response Cal Only** is selected on the previous page, click **View/Modify** to change the Source Cal settings, or click **Next>** to continue.

Otherwise, this is a [standard PNA Cal Wizard](#) page except for the following:

Power Sensor When the power sensor connector is not the same type as the DUT Port 1 connector, then for optimum accuracy, an extra cal step is required to measure the adapter that is used to connect the power sensor to the port 1 reference plane during the Source Power Cal. Specify the connector type and the Cal Kit that will be used for that step. Select **Ignored** to NOT compensate for the adapter.

Modify Cal Check, then click **Next**, to Modify Cal (Standards AND Thru Method).

View/Modify Source Cal Settings Click to launch the [Source Cal Settings](#) (for Apps) dialog.

[Learn more about IMD Calibration](#)

IMD Calibration Step 1 dialog box help



Power Level at which to perform the Port 1 Source Power Cal.

It is usually best to perform the Source Power Cal at 0 dBm because the power sensor is calibrated at that level.

However, if a component is used between the PNA source and the calibration reference plane, then adjust the power level so that the power at the sensor is about 0 dBm if possible.

[Learn more about IMD Calibration](#)

The remaining dialog pages are the same as [the standard Cal Wizard](#).

Saving Swept IMD Power Data

Swept IMD power data, Log Mag ONLY, can be saved to a *.csv file. This data type can be read by spreadsheet programs, such as Microsoft Excel. Data from the last **complete** sweep is saved to the specified *.csv file.

How to save Swept IMD (and IMDx) data

With a Swept IMD or IMDx channel active...

Using front-panel HARDKEY [softkey] buttons

1. Press **SAVE**
2. then **[Save Data As]**
3. then, **File Type = IMD Sweep Data (*.csv)**

Using a mouse with PNA Menus

1. Click **File**
2. then **Save Data As**
3. then **File Type= IMD Sweep Data (*.csv)**

Programming Commands

Notes:

- For every tone, six power levels are saved in this order: OUT Avg | OUT Lo | OUT Hi | IN Avg | IN Lo | IN Hi
- Power levels for the Main tones are always saved, regardless of the active measurement.
- All tones that are displayed are also saved. For example, any displayed 3rd order tone parameter causes the 3rd tone power levels to be saved.
- Only tone powers are saved. Calculated parameters, such as **IMD Relative to Carrier (IM)** are NOT saved.
- If calibration is turned **ON** when the file is saved, then all data is calibrated. Otherwise, raw data is saved.

Agilent Technologies,N5242A, [REDACTED]						
IDate: Tuesday, February, 24, 2009 19:53:42						
Calibration	OFF					
Num Freq F	201					
Num Tones	6					
Frequency	PwrMain	PwrMainLo	PwrMainHi	PwrMainIn	PwrMainLoIn	PwrMainHiIn
<Hz>	<LogMag	<LogMag(d	<LogMag(d	<LogMag(d	<LogMag(dBr	<LogMag(dBr
1.05E+07	-76.721	-74.1983	-79.2441	-66.6679	-21.2689	-112.067

This image shows the 6 Main tone power levels that are always saved.

In the power parameter labels, **Output** and **Avg** are implied as in the [parameter selection](#). For example: **PwrMain** = Average Output power of the main tones.

11-Sep-2013	Added ECal extrapolation
30-Oct-2012	Removed support for DC sources
3-Oct-2011	Edit description of tone power leveling
5-May-2011	Added tone power leveling and MIn/Max parameters
27-Apr-2011	Removed Copy Channels limitation
1-Apr-2011	Added note to 'Measure At'
22-Feb-2011	Enhanced 2nd order notes
12-Apr-2010	Removed some fixturing limitations
19-Nov-2009	Added X-axis point spacing (9.1)
11-Aug-2009	Added limited port mapping (A.09.00)
13-Jul-2009	Added Tone select notes
27-Feb-2009	Added data save and modified cal for IMDx
14-Aug-2008	New topic

IM Spectrum for Amplifiers (Opt 087)

Intermodulation Spectrum measurements can be made independently, or coupled with Swept IMD measurements. They are a distinct measurement class. Therefore, IM Spectrum measurements are always made in a separate channel.

- [Limitations](#)
- [Create an IM Spectrum Channel](#)
- [Select IM Spectrum Parameters](#)
- [Start the IM Spectrum Setup dialog](#)
 - [Frequency tab](#)
 - [Power tab](#)
 - [Tracking tab](#)
 - [Configure tab](#)
- [Calibration](#)

See Also

- [IM Spectrum Concepts](#)
 - [IMx Spectrum for Converters](#)
 - [Swept IMD Measurements](#)
 - [Programming commands](#)
-

Limitations with IM Spectrum

The following PNA features are NOT available in an IM Spectrum channel:

- [Unratioed receiver measurements](#) (A, B, R)
- [ECal User Characterization](#)
- [FOM](#) or [FCA](#)
- [Time Domain](#)
- [Balanced measurements](#)

- Save [Formatted Citifile](#) data.
- [Save SnP data](#).
- [External sources](#)
- [External Test Set Control](#) (Option 551)
- [Interface Control](#)
- [Port Extensions](#)
- [Integrated Narrowband](#) or [Narrowband Pulse App](#)
- [DC Meter parameters](#)

Create an IM Spectrum Channel

An IM Spectrum channel can be created independently from a Swept IMD channel or coupled with the stimulus settings of an existing Swept IMD measurement.

To create an independent IM Spectrum channel

1. On the PNA front panel, press **Meas** then **[Measurement Class]**
2. Select **IM Spectrum**, then either:
 - **OK** delete the existing measurement, or
 - **New Channel** to create the measurement in a new channel.

To create an IM Spectrum channel from an existing Swept IMD channel

1. With a Swept IMD channel active, press the **MARKER** key on the front panel.
2. Move the marker to the data point of interest. It can be moved later.
3. Press **[Marker Functions]** then **[Marker -> IM Spectrum]**

This creates or configures an existing IM Spectrum channel based upon the configuration of the Swept IMD channel.

Select IM Spectrum Parameters

How to add IM Spectrum traces

With the IM Spectrum channel active and a **Tr1 Output** trace displayed:

Using front-panel HARDKEY [softkey] buttons

1. Press **TRACES**
2. then [New Trace]
3. then select a new parameter

Using a mouse with PNA Menus

1. Click **Trace/Chan**
2. then **New Trace**
3. then select a new parameter

◀ Programming Commands ▶

New Trace - IM Spectrum -dialog box help

Check the IM Spectrum measurement to add to the display.

Choose from:

- **Output** View signals OUT of the DUT and into PNA port 2 (B receiver).
- **Input** View signals IN to the DUT (R1 receiver). Use this when measuring IM product frequencies to determine the power level of spurious signals into the DUT at those frequencies.
- **Reflection** View signals reflected off the DUT input and back into PNA port 1 (A receiver)

IM Spectrum Setup Dialog

How to start the IM Spectrum Setup dialog

To provide quicker access, use the Setup softkey. [Learn how.](#)

With an IM Spectrum measurement active:

Using front-panel HARDKEY [softkey] buttons

1. Press **SWEEP**
2. then [IM Spectrum Setup]

Using a mouse with PNA Menus

1. Click **Stimulus**
2. then **Frequency**
3. then **IM Spectrum Setup**

◀ Programming Commands ▶

Frequency tab - IM Spectrum -dialog box help

The screenshot shows a configuration window for IM Spectrum measurements. It is divided into three main sections:

- Sweep Type:** Includes radio buttons for Linear, 2nd Order Spectrum, 3rd Order Spectrum, and Nth Order Spectrum. The Nth Order Spectrum option is selected, with an Order N dropdown set to 9.
- Resolution BW:** A text field set to 600.000 kHz.
- Stimulus Settings:** Includes fields for f_c (Tone Center) at 1.00000000 GHz, Delta F at 10.000000 MHz, f₁ at 995.000000 MHz, and f₂ at 1.00500000 GHz.
- Response Settings:** Includes fields for Start Spectrum at 950.000000 MHz, Stop Spectrum at 1.05000000 GHz, Center Spectrum at 1.00000000 GHz, and Span Spectrum at 100.000000 MHz.

Note: The number of data points in an IM Spectrum channel = $3 * \text{SPAN} / \text{ResBW}$. This can NOT be changed directly.

Configures the Sweep Type and frequency range for IM Spectrum measurements.

Sweep Type

Provides several methods to tune the IM Spectrum **receivers** (NOT stimulus) to view the power spectrum of arbitrary frequency ranges or various distortion products.

Linear Allows the start/stop or center/span receiver frequencies to be set arbitrarily for f₁ and f₂. Enter the Response Settings below.

2nd Order Couples the receiver frequency range to f₁ and f₂ to provide a convenient means of observing the spectrum surrounding **ONLY** the high-side 2nd order harmonic where Center = (f₁ + f₂), Span = $3 * (f_2 - f_1)$. **Note:** The center frequency is NOT set on the main tones.

3rd Order Couples receiver frequency range to f₁ and f₂ to provide a convenient means of observing the spectrum surround the main-tones including the 3rd Order products where Center = $(f_1 + f_2) / 2$, Span = $4 * (f_2 - f_1)$.

Nth Order Couples receiver frequency range to f₁ and f₂, providing a convenient means of observing the spectrum surrounding the main-tones for an arbitrary span where Center = $(f_1 + f_2) / 2$, and Span = $N * (f_2 - f_1)$. This allows you to set the span arbitrarily, but have the center frequency track the main tone frequencies.

Resolution BW

The IM Spectrum channel utilizes a set of Gaussian filters instead of the standard PNA IF filters in order to provide similar behavior to traditional spectrum analyzers.

The comprehensive list of filters that are available for IM spectrum are: 3 MHz, 1 MHz, 600 kHz, 300 kHz, 150 kHz, 100 kHz, 60 kHz, 10 kHz, 3 kHz, and 1 kHz.

Note: The **10 kHz** Res BW filter can generate image signals which are not a product of the DUT. You can verify the integrity of a questionable signal by switching to the 3 kHz or 60 kHz filter and looking for the image signal in the same location.

Not all of these filters are available for all measurements. Narrower filters are available for use with narrower frequency spans, and wider filters are available for wider spans.

Stimulus Settings

Allows configuration of the main-tone frequencies. Available ONLY when Tracking is OFF.

- **fc (main-tone center frequency)** = $(f1 + f2) / 2$
- **DeltaF (main-tone frequency separation)** = $(f2 - f1)$.
- **f1** = Low-side main-tone frequency
- **f2** = High-side main-tone frequency

Response Settings

Allows configuration of the frequency range to sweep the receivers in terms of start/stop or center/span. Available for Linear sweep types ONLY.

- **Start Spectrum** = first frequency point of the power spectrum sweep
- **Stop Spectrum** = last frequency point of the power spectrum sweep
- **Center Spectrum** = $(\text{Start Spectrum} + \text{Stop Spectrum}) / 2$
- **Span Spectrum** = $(\text{Stop Spectrum} - \text{Start Spectrum})$

Power tab - IM Spectrum -dialog box help

Power On (All Channels)

DUT Input
Input Port: Port 3
Source Attenuator: 0 dB
Receiver Attenuator: 0 dB

DUT Output
Output Port: Port 4
Source Attenuator: 10 dB
Receiver Attenuator: 0 dB

Tone Powers
 Coupled Tone Powers ALC On

Power Leveling
 Set Input Power
 Set Input Power, receiver leveling
 Set Input Power, equal tones at output
 Set Output Power, receiver leveling

	f1 Power	f2 Power
Fixed	-20.00 dBm	-20.00 dBm
Start	-20.00 dBm	-20.00 dBm
Stop	-10.00 dBm	-10.00 dBm
Step	0.050 dB	0.050 dB

External combiner required for measurements at port 3 and 4

Path Configuration...

[Learn about this dialog](#)

Tracking tab - IM Spectrum -dialog box help

Tracking Enable Tracking Channel: 3

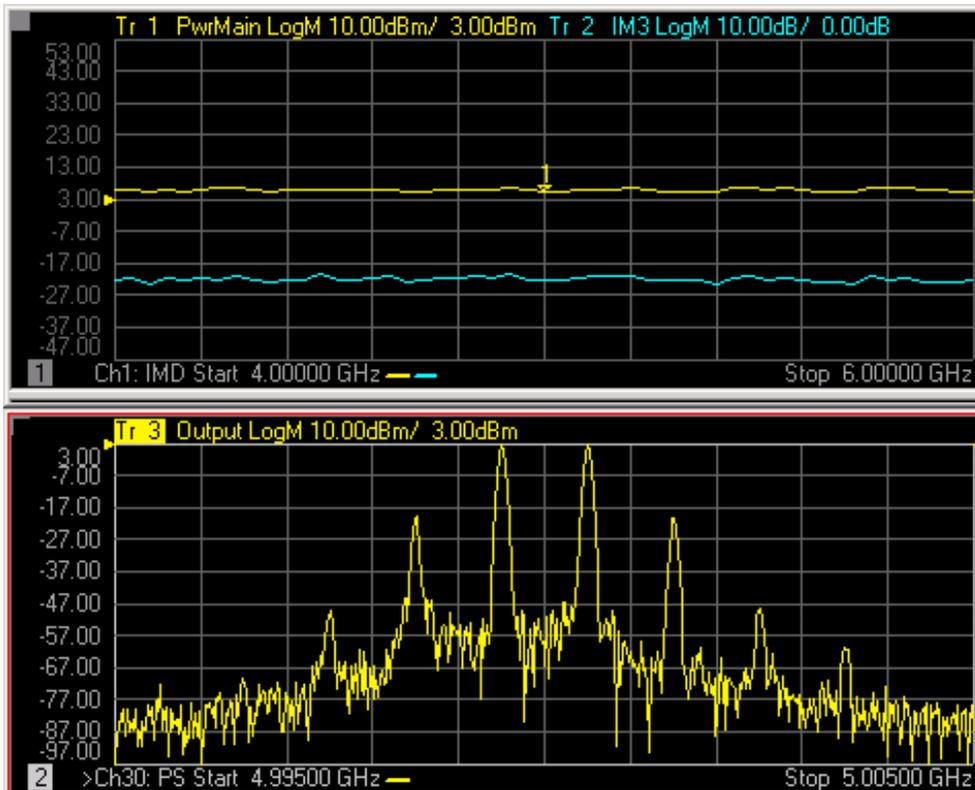
Step Mode
 Automatic Step
Automatically measure the spectrum at each stimulus point in the Swept IMD

Manual Step Stimulus Points: 1
Measure the spectrum at stimulus point #1 of 201 in the Swept IMD channel

These selections are NOT available when there is no Swept IMD channel.

Allows the IM Spectrum channel to use (track) the **stimulus** (Main Tone) settings of an existing Swept IMD channel.

Tracking Enable Check to use the frequency and power stimulus settings from the specified IMD channel. When enabled, stimulus settings on the Frequency Tab are disabled and ALL stimulus settings, such as frequencies, power and attenuator settings, and calibration, are copied from the Swept IMD channel to the IM Spectrum channel.



IM Spectrum trace (bottom) tracking with Swept IMD traces (top)

- In the **top** Swept IMD window the main tones are swept from 4 GHz to 6 GHz with some specified delta F tone separation. The marker is on the center data point at **5 GHz**.
- The **bottom** IM Spectrum window center frequency is the same as the above marker: **5 GHz**, but it has a much narrower frequency range of +/- 500 kHz. The IM Spectrum channel sets the receiver to see the two main tones, plus the third, fifth, and seventh-order products.

Step Mode

When tracking is enabled, set the method by which the IM spectrum measurement tracks the IMD channel.

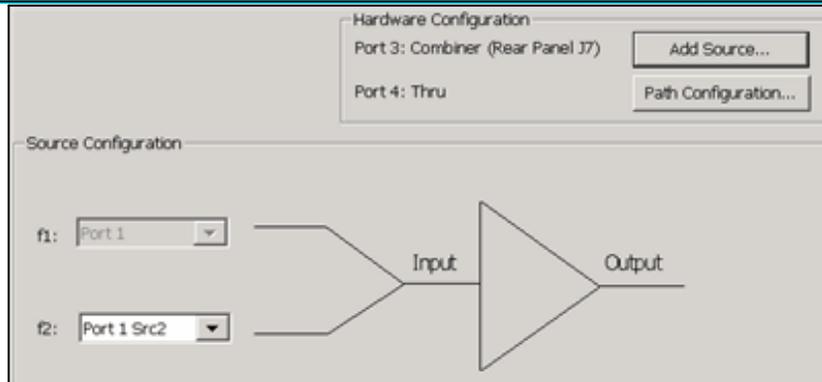
Manual Step When selected, IM Spectrum measurements occur at only the specified IMD channel data point. **Stimulus Point** specifies the data point, by number, in the Swept IMD channel which has the stimulus settings to use for the IM Spectrum sweep.

Automatic Step When selected, causes the IM Spectrum channel to sequentially setup each of the stimulus conditions through which the Swept IMD channel sweeps the DUT. Each sweep of IM Spectrum is performed using the next set of stimulus conditions.

For example, in the above image, the first Swept IMD data point is at 4 GHz. The first IM

Spectrum sweep uses a center frequency of 4 GHz. The following IM Spectrum sweep would be at the second Swept IMD data point or 4.01 GHz, and so forth. After the last data point in the sweep is reached, the IM Spectrum channel begins again at the first Swept IMD data point. The only indication that Tracking is enabled is in Automatic mode, you can see the center frequency increment with each IM Spectrum sweep.

Configure tab- IM Spectrum Setup -dialog box help



To accommodate single-source PNA models, an external source can be used for the RF2 tone.

Learn how to [configure an external source and combiner](#) to make Swept IMD, IMDx, IM Spectrum, and IM Spectrum for Converters measurements.

RF1 Always uses PNA internal source 1.

RF2 Available for selection when an external source is configured and Active.

Add Source Click to configure an external source using the [External Source Configuration dialog](#).

Path Configuration Click to launch the [Path Configuration dialog](#) (PNA-X models only).

Calibration

A calibration corrects the IM Spectrum source and receiver power level accuracy of the displayed Tones.

A calibration of the IM Spectrum channel is NOT performed using the Calibration wizard. An IM Spectrum channel is calibrated by applying a Cal Set in one of the following ways:

- A cal that was used on a Swept IMD channel. The Cal Set can be applied to the IM Spectrum channel using

the Manage Cal Sets dialog ([Learn how](#)) or from the Marker =>IM Spectrum softkey ([Learn how](#)). However, a Swept IMD channel with [Sweep Type = Power Sweep](#) can NOT be applied to a IM Spectrum channel. This is because a Cal Set for power sweep contains only a CW frequency and the IM Spectrum channel requires a swept frequency range. Zero Span is not supported in an IM Spectrum channel.

- A Source Power and Receiver Cal Set from a standard channel calibration. This can be a full calibration, but must include a source power cal for the source port (1) and receiver cal for the B receiver. Only one source can be corrected. [Learn how to apply a standard Cal Set.](#)

[See Swept IMD Calibration](#)

Last Modified:

3-Aug-2012	No DC control
11-Oct-2011	Modified calibration content
27-Apr-2011	Removed Copy Channels limitation
16-Sep-2009	Fixed calibration
11-Aug-2009	Added limited port mapping
18-Aug-2008	MX New topic

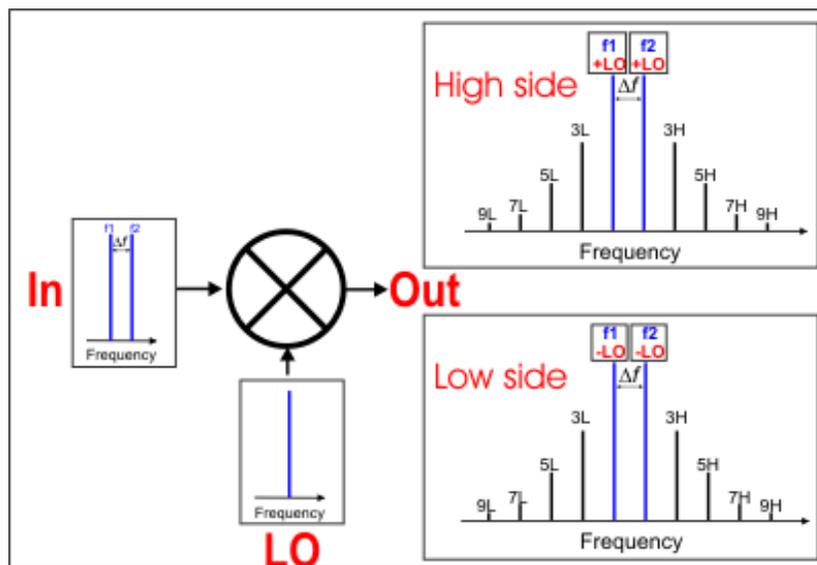
Swept IMD for Converters (IMDx)

- [IMDX Concepts](#)
- [Requirements and Limitations](#)
- [How to Configure your Hardware](#)
- [Create a Swept IMDx Measurement](#)
- [Valid Mixer Configuration / Sweep Type Combinations](#)
- [Setup Dialog](#)
- [How to specify IMDx Parameters](#)
- [IMDx Calibration](#)
- [Saving IMDx Data](#) (Swept IMD topic)

Other IMD Topics

IMDx Concepts

Conceptually, Swept IMD for Converters (IMDx) is like [IMD for Amplifiers](#), except that there are two sets of products on the DUT output.



This image shows the simplest measurement configuration: the Input, LO, and Output are all CW frequencies.

- DUT **INPUT** - two fundamental tones (f1 and f2)
- DUT **LO** - a single frequency.
- DUT **OUTPUT** - two sets of frequencies

With IMDx, you can view EITHER the High side or Low side products; NOT BOTH.

Make this selection on the [Mixer Frequency tab](#).



High side (Input **PLUS** LO and all ordered products)

Low side (Input **MINUS** LO and all ordered products)

Requirements and Limitations

Swept IMDx requires [Swept IMD](#) (Opt 087) and [FCA](#) (either Opt 082 or 083).

- When using [Integrated Pulse application](#), the IF Filter setting for the relevant receiver must be changed to 'Wide'. [Learn how](#). The default IF Filter setting in Swept IMD is 'Narrow' in order to avoid spurs and harmonics. The IF Attenuator setting (on the same dialog) may also require adjusting.

The following PNA features are **NOT** available with Swept IMDx:

- Analog Sweep ([Stepped sweep](#) mode only)
- [Log frequency](#) sweeps
- [Unratioed receiver measurements](#) (A, B, R)
- [ECal User Characterization](#)
- [Time Domain](#)
- [Balanced measurements](#)
- Save [Formatted Citifile](#) data.
- [Save SnP data](#).
- [Interface Control](#)
- [Port extensions](#)
- [Some Fixturing Features](#)
- [External Test Set Control](#) (Option 551)
- [Integrated Narrowband](#) or [Narrowband Pulse App](#)

- Independent IFBW, Power Levels, or Sweep Time in a [segment table](#) is NOT supported.
- [External DC Sources](#) (DC Meters ARE supported).
- [See Frequency Limitations](#)

Note: Beginning with A.09.00, **Embedded LO** measurements are allowed in IMDx and IMSpectrum. Configure the measurement as you would with SMC. [Learn how.](#)

How to Configure your Hardware

The PNA is extremely versatile, and can be configured in many ways to make IMDx measurements. While not all conceivable configurations are documented here, a few of the most common examples are provided to show the basic concepts.

DUT Configuration

- The DUT Input must be connected to PNA Port 1 which supplies the f1 and f2 tones.
- The DUT Output must be connected to PNA Port 2 which uses the PNA B receiver.
- See LO Source configuration below.

Source Configuration

Three sources are required to make IMDx measurements: Two sources are PNA internal; the third is an external source.

- **F1 tone** - Must come out PNA Port 1 Default is Src 1
- **F2 tone** - Must come out PNA Port 1 through the internal combiner. This source can come from an internal or external source. Default is the internal Src 2.
- **LO** - Can come from internal or external source. Default is Not controlled, set to 0 Hz.
 - If using the internal Src 2 source, the f2 tone must come from an external source through the rear-panel.
 - If using an external source, it can be connected directly to the DUT, or through the PNA Port 3 or Port 4 using the Path Configurator.

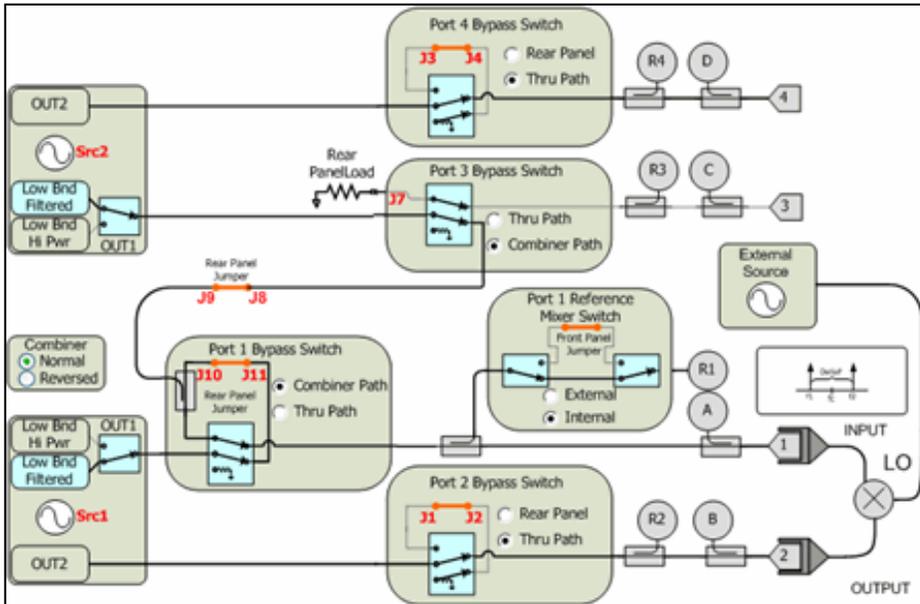
Configuration Examples

Three configurations are shown below to illustrate how to use the [Path Configurator](#) to manually make switch settings.

IMDx and IMx Spectrum channels are ALWAYS configured using the default configuration switch

settings shown in the image below. This is NOT the same default configuration that is used for an S-parameter channel. A manual switch setting is required whenever a custom configuration is used.

The manual switch setting, which must be done every time an IMDx or IMx Spectrum channel is created, can be saved using the **Store** button in the Path Configurator. Then save the entire IMDx setup as an [Instrument State](#). This will load the custom Path Configuration when the Instrument State is recalled.



1. Default Configuration No manual switching required.

- f1 - Internal Src1
- f2 - Internal Src2 through combiner in Normal
- LO - Connect external source directly to the DUT LO.

2. External LO though Port 3 or Port 4 (4-port PNA only)

Use this configuration to monitor LO power using R3 or R4. (Future parameters)

- f1 - Internal Src1
- f2 - Internal Src2 through combiner in Normal
- LO - Connect external source through the rear-panel (J7 for Port 3; J3 for Port 4).
 - Connect the DUT LO to PNA Port 3 or Port 4.
 - For Port 3, NO switching is required.

- For Port 4, switch Port 4 Bypass Switch to **Rear Panel**.

3. Internal LO through Port 3 or Port 4; External f2 through rear-panel (4-port PNA only)

This configuration is commonly used for FCA measurements where the internal second source is used as the LO.

- **f1** - Internal Src1
- **f2** - External f2 through rear-panel J9. No switching required.
- **LO** - Internal Src2 through:
 - Port 3 Switch Port 3 Bypass Switch to **Thru Path..** Connect the DUT LO to PNA Port 3
 - Port 4 Switch Port 4 Bypass Switch to **Thru Path.** Connect the DUT LO to PNA Port 4

Create a Swept IMDx Measurement

1. On the PNA front panel, press **Meas** then **[Measurement Class]**
2. Select **Swept IMD Converters**, then either:
 - **OK** delete the existing measurement, or
 - **New Channel** to create the measurement in a new channel.
3. A Swept IMDx measurement is displayed. To select additional parameters to display, click **Response**, then **Measure**, then select a parameter from the list.

How to start the Swept IMDx Setup dialog

To provide quicker access, use the Setup softkey. [Learn how.](#)

Using front-panel HARDKEY [softkey] buttons

1. Press **FREQ**
2. then **[Swept IMDx Setup]**

Using a mouse with PNA Menus

1. Click **Stimulus**
2. then **Frequency**
3. then **Swept IMDx Setup**

Programming Commands

Valid Mixer Configuration / Sweep Type Combinations

Tips

Although you will soon become comfortable navigating these tabs, at first it may be best to complete the dialog in the following order:

1. On the [Tone Frequency tab](#), set the Sweep Type and Tone Frequencies.
 2. On the [Mixer Frequency tab](#), set a valid mixer configuration.
- Input center / fixed frequency CAN be set on both the [Tone Frequency tab](#) and the [Mixer Frequency tab](#). When you set one, the other is updated automatically.

The following are the **Valid Mixer Configurations** in table format:

Sweep Type	Input	LO	Output
Swept Input (Fixed Tones)			
Sweep fc	Swept	Swept	Fixed
	Swept	Fixed	Swept
All Ranges Fixed			
CW	Fixed	Fixed	Fixed
Sweep Delta F			
(Tone) Power Sweep			
LO Power Sweep			
Fixed Input / Swept LO & Output			
CW	Fixed	Swept	Swept

For determining a valid mixer configuration with 2 LOs, one Fixed LO and one Swept is equivalent to having a single-stage Swept LO.

If you create an invalid Sweep Type / Mixer Configuration, a red message appears like the following:

ERROR: Input range must be swept in Linear sweep mode

- If this occurs, change the **Sweep Type** on the [Tone Frequency](#) tab.
- [See other rules for configuring a mixer.](#)
- The following is an explanation of the table:

SWEPT Input - Sweeps the center frequency of the tones, but the tone spacing remains fixed.

On the [Tone Frequency](#) tab, select **Sweep fc**

- Either the LO or Output MUST also be swept.
- On the Mixer Frequency tab, select Start/Stop, or Center/Span for each range to be swept.

FIXED Input - The center frequency of the tones is fixed.

On the Tone Frequency tab select **Fixed** for the following sweep type:

- **CW** - Tones do NOT change. The LO and Output frequencies CAN be swept.

All Ranges FIXED - The center frequency of the tones is fixed.

On the Tone Frequency tab select **Fixed** for the following sweep types:

- **Sweep Delta F** - Tone separation changes.
- **Power Sweep** - Tone power changes.
- **LO Power Sweep** - The LO power is swept.
- **CW** - Tones do NOT change.

Setup Dialog

The following tabs are shared with the Swept IMD Setup dialog:

- [Tone Frequency tab](#)
- [Tone Power tab](#)

The following tabs are shared with all Mixer / Converter Applications

- [Mixer Frequency tab](#)
- [Mixer \(LO\) Power tab](#)
- [Mixer Setup tab](#) (NOT shared)

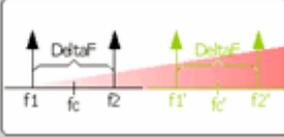
Tone Frequency tab - Swept IMDx Setup -dialog box help

Sweep Type

- Sweep f_c
- Sweep DeltaF
- Power Sweep
- CW
- LO Power Sweep

X-Axis Display

Annotation:



Sweep Settings

Start f _c :	<input type="text" value="10.500000 MHz"/>	Stop f _c :	<input type="text" value="26.499500000 GHz"/>
Center f _c :	<input type="text" value="13.255000000 GHz"/>	Span f _c :	<input type="text" value="26.489000000 GHz"/>
Fixed DeltaF:	<input type="text" value="1.000000 MHz"/>	Main Tone IFBW:	<input type="text" value="1.000 kHz"/>
Number Of Points:	<input type="text" value="201"/>	IM Tone IFBW:	<input type="text" value="1.000 kHz"/>

Reduce IF BW at Low Frequencies

[Learn about this dialog.](#)

Tone Power tab - Swept IMDx Setup -dialog box help

Power On (All Channels)

DUT Input

Input Port:

Source Attenuator:

Receiver Attenuator:

DUT Output

Output Port:

Source Attenuator:

Receiver Attenuator:

Tone Powers

Coupled Tone Powers ALC On

	f ₁ Power	f ₂ Power
Fixed	<input type="text" value="-20.00 dBm"/>	<input type="text" value="-20.00 dBm"/>
Start	<input type="text" value="-20.00 dBm"/>	<input type="text" value="-20.00 dBm"/>
Stop	<input type="text" value="-10.00 dBm"/>	<input type="text" value="-10.00 dBm"/>
Step	<input type="text" value="0.050 dB"/>	<input type="text" value="0.050 dB"/>

Power Leveling

- Set Input Power
- Set Input Power, receiver leveling
- Set Input Power, equal tones at output
- Set Output Power, receiver leveling

External combiner required for measurements at port 3 and 4

[Learn about this dialog.](#)

Mixer Frequency tab - Swept IMDx Setup -dialog box help

[Learn about this dialog](#)

Mixer Setup tab - Swept IMDx Setup -dialog box help

Converter Stages Choose from 1 or 2 stage DUT (# of LOs).

Converter Model

Learn how to [configure an external source and combiner](#) to make Swept IMD and IMDx measurements.

f1 Always uses PNA internal source 1.

(DUT Input) Port N (**Red** circle in image) Normally Port 1 is used. This selection is available to allow the two tones to be routed through port 3 on 4-port PNA models. [Learn more](#).

f2 Select a source to be used for the f2 tone. This selection is available when an external source is

configured and the **Active** box is checked on the [External Source Configuration dialog](#).

LO1 (and LO2 for 2-stage DUTs) Select the source to use for the specified LO. Available for selection when an external source is configured and the **Active** box is checked on the [External Source Configuration dialog](#).

(DUT Output) Port N Available for selection on 4-port PNA models. Select the PNA port to connect to the DUT Output.

Fractional Multipliers - [Learn more](#).

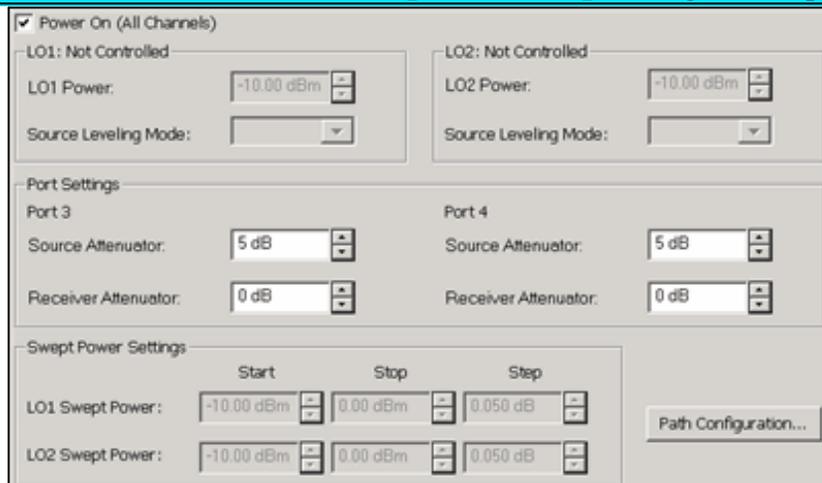


Hardware Configuration

Add Source Click to configure an external source using the [External Source Configuration dialog](#).

Path Configuration Click to launch the [Path Configuration dialog](#) (PNA-X models only).

Mixer (LO) Power tab - Swept IMD Setup -dialog box help



Power On (All Channels)

LO1: Not Controlled
LO1 Power: -10.00 dBm
Source Leveling Mode: [v]
LO2: Not Controlled
LO2 Power: -10.00 dBm
Source Leveling Mode: [v]

Port Settings

Port 3
Source Attenuator: 5 dB
Receiver Attenuator: 0 dB

Port 4
Source Attenuator: 5 dB
Receiver Attenuator: 0 dB

Swept Power Settings

	Start	Stop	Step
LO1 Swept Power:	-10.00 dBm	0.00 dBm	0.050 dB
LO2 Swept Power:	-10.00 dBm	0.00 dBm	0.050 dB

Path Configuration...

[Learn about this dialog](#)

How to add Swept IMDx Parameters

Using front-panel HARDKEY [softkey] buttons

1. Press TRACES
2. then [New Trace]
3. then select a parameter

Using a mouse with PNA Menus

1. Click Trace/Chan
2. then New Trace
3. then select a parameter

Programming Commands

IMDx New Trace dialog box help

Param. Name	Type	Tone Select	Order	Measure At
<input type="checkbox"/> PwrMain	Tone Power	Avg	1	OUT OUT
<input type="checkbox"/> IM3	IMD Relative to Carrier	Avg	3	OUT OUT
<input type="checkbox"/> OIP3	Output Referred Intcpt. Pt.	Avg	3	OUT OUT
<input type="checkbox"/> IIP3	Input Referred Intcpt. Pt.	Avg	3	OUT OUT
<input type="checkbox"/> Pwr2	Tone Power	Avg	2	OUT OUT

This dialog is shared with Swept IMD for Amplifiers.

[Learn about this dialog.](#)

IMDx Calibration

Calibration for IMDx is exactly the same as [calibration for IMD](#) with the following exception:

- You can choose to perform a source power cal of the LO source. If the LO is a fixed frequency, this step is performed very fast.

The results of an IMDx calibration are very similar to the results that are achieved from an [SMC](#)

[calibration.](#)

Last Modified:

1-Feb-2013	Mixer setup edits
27-Apr-2011	Removed Copy Channels limitation
27-Sep-2010	Merged with Swept IMD and Mixer tabs
12-Apr-2010	Removed some fixturing limitations
11-Aug-2009	Added limited port mapping
13-Jul-2009	Added 2nd order notes
2-Feb-2009	New topic

IMx Spectrum for Converters

IMx Spectrum measurements are a distinct measurement class and therefore always made in a separate channel. This topic discusses all aspects of an IMx Spectrum measurement.

- [Limitations](#)
- [How an IMx Spectrum Channel Works](#)
- [Create an IMx Spectrum Channel](#)
- [Select IMx Spectrum Parameters](#)
- [Start the IMx Spectrum Setup dialog](#)
 - [Frequency tab](#)
 - [Tone Power tab](#)
 - [Mixer Frequency tab](#) (separate topic)
 - [Mixer Power tab](#) (separate topic)
 - [Mixer Setup tab](#)
- [Calibration](#)

See Also

- [Swept IMD Measurements](#)
 - [Programming commands](#)
-

Limitations

IMx Spectrum requires IMD (Opt 087) and [FCA](#) (either Opt 082 or 083).

Note:

The **minimum** frequency of this application is **40 MHz**.(10 MHz plus 30 MHz).

The **maximum** frequency of this application is the MAX frequency of the PNA **MINUS 30 MHz**.

This is due to the use of the various IF frequencies.

The following PNA features are NOT available in an IMx Spectrum channel:

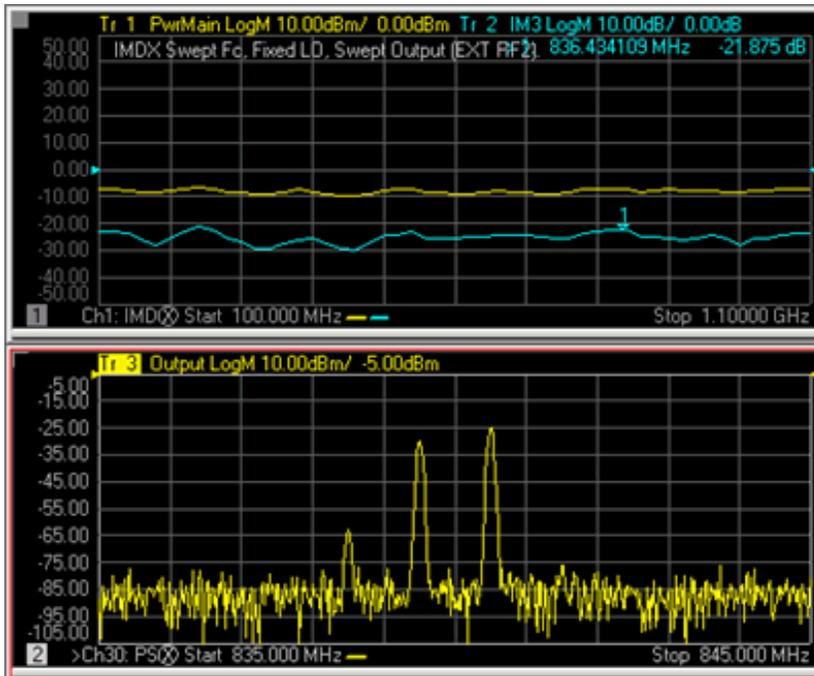
- [Unratioed receiver measurements](#) (A, B, R)
- [ECal User Characterization](#)
- [FOM](#) or [FCA](#)
- [Time Domain](#)
- [Balanced measurements](#)
- Save [Formatted Citifile](#) data.
- [Save SnP data](#).
- [External sources](#)
- [External Test Set Control](#) (Option 551)
- [Interface Control](#)
- [Port Extensions](#)
- [Fixturing Simulator](#)
- [Integrated Narrowband](#) or [Narrowband Pulse App](#)

Note: Opt 084 ([Embedded LO measurements](#)) is NOT necessary to make an IMx Spectrum measurement on converters with an embedded LO. This is because the embedded LO can be slightly off-frequency and the output tones will still be visible on the IMxSpectrum display. You may have to select **Linear** sweep type, then set the response frequency span wide enough to allow for a more off-frequency embedded LO. [Learn how](#).

How an IMx Spectrum for Converters Channel Works

Before reading this topic, you should become familiar with [IMD Concepts](#).

The IMx Spectrum channel provides a traditional Spectrum Analyzer view of the intermodulation distortion behavior of a converter output. Unlike the Swept IMDx channel, the main tones (F1 and F2) are fixed while the receiver is swept over a frequency range of interest in order to generate a display as shown below.



IMx Spectrum trace (bottom) with Swept IMDx traces (top)

A typical Spectrum Analyzer does NOT have a signal source. This one does. The signal source, or stimulus, settings for the F1 and F2 main tones can be set in either the IMx Spectrum channel or the Swept IMDx channel. These settings include the frequencies and power levels of the main tones.

Receiver Settings

Although the stimulus settings can be set in either the IMx Spectrum channel or the Swept IMDx channel, the receiver settings are set ONLY in the Frequency tab of the [IMx Spectrum dialog](#). These settings include how many tone products to view, which determines the center and frequency span and the resolution bandwidth. You can also set a Linear sweep type, then enter an arbitrary receiver frequency range.

You can choose from several resolution bandwidths which also determines the number of data points used in the channel. The higher the Res BW, the fewer the number of data points.

The IMx Spectrum channel performs multiple measurements for each data point in order to reject unwanted images which are generated by the PNA internally. This provides a high degree of confidence that signals captured in a trace are real and are not spurious responses generated in the measurement process.

Differences from IM Spectrum for Amplifiers

An IMx Spectrum for Converters channel has the following important differences from an [IM Spectrum channel for Amplifiers](#).

- In an IMx Spectrum channel, **all converter frequencies are fixed.** (Input, Output, LO, Tone Spacing, and Tone Power).
- The IMx Spectrum channel **CANNOT track** with the IMDx channel, as it can in IMD for Amplifiers.

Create an IMx Spectrum Channel

An IMx Spectrum channel can be created independently, or from an existing Swept IMDx channel.

To create an independent IMx Spectrum channel

1. On the PNA front panel, press **Meas** then **[Measurement Class]**
2. Select **IMx Spectrum**, then either:
 - **OK** delete the existing measurement, or
 - **New Channel** to create the measurement in a new channel.

To create an IMx Spectrum channel from an existing Swept IMD channel

This creates or configures an existing IMx Spectrum channel based upon the configuration of the Swept IMD channel at the Marker frequency.

1. With a Swept IMD channel active, press the **MARKER** key on the front panel.
2. Move the marker to the data point of interest. It can be moved again later.
3. Press **[Marker Functions]** then **[Marker -> IMx Spectrum]**

Select IMx Spectrum Parameters

How to add IMx Spectrum traces

With the IMx Spectrum channel active and a **Tr1 Output** trace displayed:

Using front-panel HARDKEY [softkey] buttons

1. Press **TRACES**
2. then **[New Trace]**
3. then select a new parameter

Using a mouse with PNA Menus

1. Click **Trace/Chan**
2. then **New Trace**
3. then select a new parameter

Programming Commands

New Trace - IMx Spectrum -dialog box help

Output Port 2, Incident

View signals OUT of the DUT and into PNA port 2 (B receiver).

Note: Only viewing the OUTPUT tones is supported.

IMx Spectrum Setup Dialog

How to start the IMx Spectrum Setup dialog

To provide quicker access, use the Setup softkey. [Learn how.](#)

With an IMx Spectrum measurement active:

Using front-panel HARDKEY [softkey] buttons

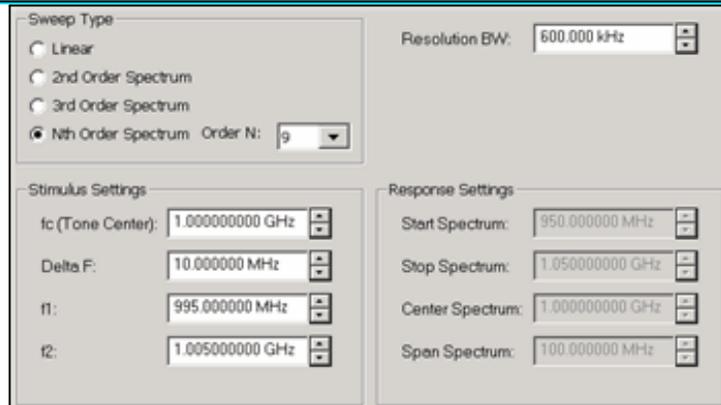
1. Press **SWEEP**
2. then **[IMx Spectrum Setup]**

Using a mouse with PNA Menus

1. Click **Stimulus**
2. then **Frequency**
3. then **IMx Spectrum Setup**

[Programming Commands](#)

Frequency tab - IMx Spectrum -dialog box help



Sweep Type

- Linear
- 2nd Order Spectrum
- 3rd Order Spectrum
- Nth Order Spectrum Order N:

Resolution BW:

Stimulus Settings

fc (Tone Center):

Delta F:

f1:

f2:

Response Settings

Start Spectrum:

Stop Spectrum:

Center Spectrum:

Span Spectrum:

[Learn about this dialog](#) and [about *.mxr files.](#)

Tone Power tab - IMxSpectrum -dialog box help

Power On (All Channels)

DUT Input
Input Port:
Source Attenuator:
Receiver Attenuator:

DUT Output
Output Port:
Source Attenuator:
Receiver Attenuator:

Tone Powers
 Coupled Tone Powers ALC On

Power Leveling
 Set Input Power
 Set Input Power, receiver leveling
 Set Input Power, equal tones at output
 Set Output Power, receiver leveling

	f1 Power	f2 Power
Fixed	<input type="text" value="-20.00 dBm"/>	<input type="text" value="-20.00 dBm"/>
Start	<input type="text" value="-20.00 dBm"/>	<input type="text" value="-20.00 dBm"/>
Stop	<input type="text" value="-10.00 dBm"/>	<input type="text" value="-10.00 dBm"/>
Step	<input type="text" value="0.050 dB"/>	<input type="text" value="0.050 dB"/>

External combiner required for measurements at port 3 and 4

Path Configuration...

[Learn about this dialog](#) and [about *.mxr files](#).

Mixer Frequency tab - IMx Spectrum -dialog box help

Mixer Frequency

Input:

LO1: Input > LO

IF: +
 -

LO2: IF1 > LO2

Output: +
 -

[Learn about this dialog](#) and [about *.mxr files](#).

Mixer (LO) Power tab - IMx Spectrum -dialog box help

Power On (All Channels)

LO1: Not Controlled
LO1 Power: -10.00 dBm
Source Leveling Mode: []

LO2: Not Controlled
LO2 Power: -10.00 dBm
Source Leveling Mode: []

Port Settings

Port 3
Source Attenuator: 5 dB
Receiver Attenuator: 0 dB

Port 4
Source Attenuator: 5 dB
Receiver Attenuator: 0 dB

Swept Power Settings

	Start	Stop	Step
LO1 Swept Power:	-10.00 dBm	0.00 dBm	0.050 dB
LO2 Swept Power:	-10.00 dBm	0.00 dBm	0.050 dB

Path Configuration...

[Learn more about this dialog](#) and [about *.mxr files](#).

Mixer Setup tab - IMx Spectrum -dialog box help

Converter Stages: 2

Hardware Configuration
Port 3: Combiner (Rear Panel J7) Add Source ...
Port 4: Thru Path Configuration...

Converter Model: Single Stage

f1: Port 1
f2: Port 1 Src2

Port 1

Port 4

LO1: Not controlled LO2: PSG

To accommodate single-source PNA models, an external source can be used for the RF2 tone.

Learn how to [configure an external source and combiner](#) to make Swept IMD and IMDx measurements.

Converter Stages Choose from 1 or 2 stage DUT (# of LOs).

Converter Model

f1 Always uses PNA internal source 1.

(DUT Input) Port N Available for selection on 4-port PNA models. Select the PNA port (1 or 3) to connect to the DUT Input.

f2 Available for selection when an external source is configured and Active.

LO1 (and LO2 for 2-stage DUTs) Select the source to use for the specified LO. Available for selection when an external source is configured and Active.

(DUT Output) Port N Available for selection on 4-port PNA models. Select the PNA port to connect to the DUT Output.

Hardware Configuration

Add Source Click to configure an external source using the [External Source Configuration dialog](#).

Path Configuration Click to launch the [Path Configuration dialog](#) (PNA-X models only).

Calibration

A calibration of the IMx Spectrum channel is NOT performed using a calibration wizard.

An IMx Spectrum channel is calibrated from a Cal Set that is used on a Swept IMDx channel. The Cal Set can be applied to the IMx Spectrum channel using the Manage Cal Sets dialog ([Learn how](#)) or from the Marker =>IMx Spectrum softkey ([Learn how](#)).

However, a Swept IMDx channel with [Sweep Type = Power Sweep](#) can NOT be applied to a IMx Spectrum channel. This is because a Cal Set for power sweep contains only a CW frequency and the IMx Spectrum channel requires a swept frequency range. Zero Span is not supported in an IMx Spectrum channel.

A calibrated IMx Spectrum trace corrects the source and receiver power level accuracy of the displayed Tones.

[See Swept IMD Calibration](#)

[Learn how to apply a Cal Set to the IMx Spectrum channel](#)

Last Modified:

9-Jul-2013	Added min freq limit
30-Nov-2011	Added Opt 084 note
9-May-2011	Edited Mixer Setup
27-Apr-2011	Removed Copy Channels limitation
16-Aug-2010	Diverted to common Mixer topic
16-Sep-2009	Fixed Calibration
11-Aug-2009	Added limited port mapping
2-Feb-2009	New topic

Time Domain

Time Domain allows you to view a device response as a function of time. The following are discussed in this topic:

- [Overview](#)
- [How the PNA Measures in the Time Domain](#)
- [Calibration for Time Domain](#)
- [Transmission Measurements](#)
- [Measurement Response Resolution](#)
- [Measurement Range and Alias Responses](#)
- [How to make Time Domain Settings](#)
- [Gating](#)
- [Window Settings](#)

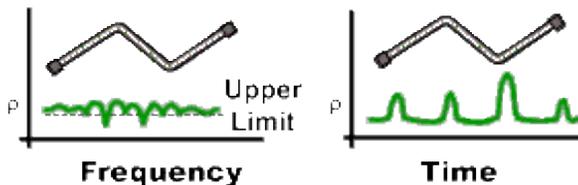
Note: Time Domain measurements are only available on PNAs with Option 010. See [PNA Options](#)

See the updated **App Note:** [Time Domain Analysis Using a Network Analyzer](#).

Overview

In normal operation, the PNA measures the characteristics of a test device as a function of frequency. With Time Domain (opt 010), the frequency information is used to calculate the inverse Fourier transform and display measurements with time as the horizontal display axis. The response values appear separated in time, allowing a different perspective of the test device's performance and limitations.

The graphic below compares the same cable reflection measurement data in both the frequency and time domain. The cable has two bends. Each bend creates a mismatch or change in the line impedance.



- The frequency domain S11 measurement shows reflections caused by mismatches in the cable. It is impossible to determine where the mismatches physically occur in the cable.

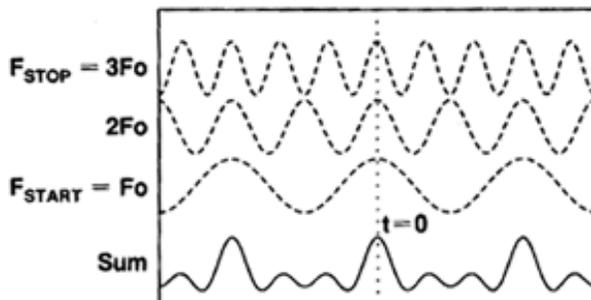
- The time domain response shows both the location and the magnitude of each mismatch. The responses indicate that the second cable bend is the location of a significant mismatch. This mismatch can be [gated out](#), allowing you to view the frequency domain response as if the mismatch were not present. Distance Markers can be used to pinpoint the distance of the mismatch from the reference plane.

How the PNA Measures in the Time Domain

Time domain transform mode simulates traditional Time-Domain Reflectometry (TDR), which launches an impulse or step signal into the test device and displays the reflected energy on the TDR screen. By analyzing the magnitude, duration, and shape of the reflected waveform, you can determine the nature of the impedance variation in the test device.

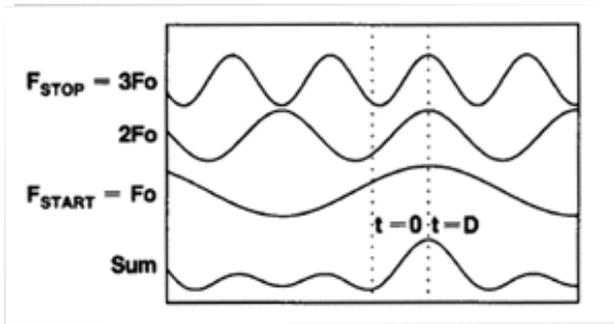
The PNA does not launch an actual incident impulse or step. Instead, a Fourier Transform algorithm is used to calculate time information from the frequency measurements. The following shows how this occurs.

A single frequency in the time domain appears as a sine wave. In the following graphic, as we add the fundamental frequency (F_0), the first harmonic ($2F_0$), and then the second harmonic ($3F_0$), we can see a pulse taking shape in the Sum waveform. If we were to add more frequency components, the pulse would become sharper and narrower. When the PNA sends discrete frequencies to the test device, it is in effect, sending individual spectral pieces of a pulse separately to stimulate the test device.

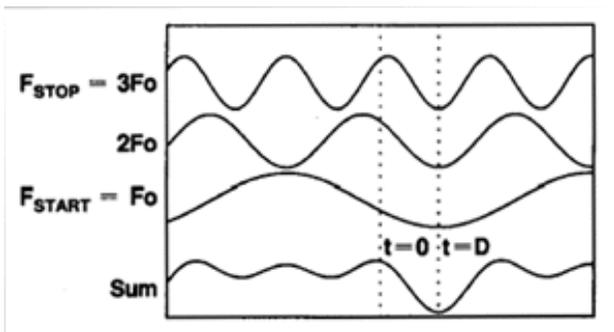


During an S11 reflection measurement, these incident signals reflect from the test device and are measured at the A receiver. This is when the time domain transform calculations are used to add the separate spectral pieces together.

For example, consider a short length of cable terminated with an open. All of the power in the incident signal is reflected, and the reflections are 'in-phase' with the incident signal. Each frequency component is added together, and we see the same pattern as the simulated incident would have looked (above). The magnitude of the reflection is related to the impedance mismatch and the delay is proportional to the distance to the mismatch. The x-axis (time) scale is changed from the above graphic to better show the delay.



Alternately, the same cable terminated with a short also reflects all of the incident power, but with a phase shift of 180 degrees. As the frequency components from the reflection are added together, the sum appears as a negative impulse delayed in time.



Calibration for Time Domain

For simplicity, we have discussed incident signals reflecting off discontinuities in the test device. By far the most common network analyzer measurement to transform to time domain is a [ratioed S11](#) measurement. An S11 reflection measurement does not simply display the reflections measured at the A receiver - it displays the ratio (or difference) of the A receiver to the Reference receiver. In addition, the S11 measurement can also be calibrated to remove [systematic errors](#) from the ratioed measurement. This is critical in the time domain as the measurement plane, the point of calibration, becomes zero on the X-axis time scale. All time and distance data is presented in reference to this point. As a result, both magnitude and time data are calibrated and very accurate.

The following shows where the time domain transform occurs in the PNA data flow: (see [Data Access Map](#))

1. Acquire raw receiver (A and R1) data
2. Perform ratio (A/R1)
3. Apply calibration
4. Transform data to time domain

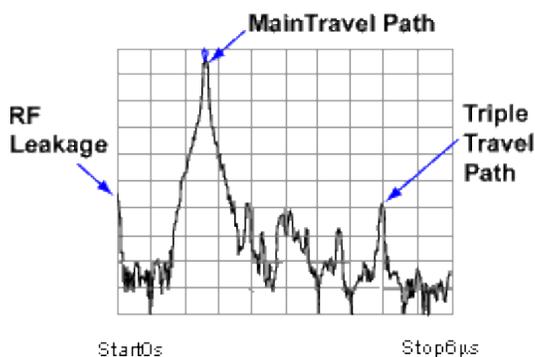
5. Display results

Therefore, although a time domain trace may be displayed, a calibration is always performed and applied to the frequency domain measurement which is not displayed.

Transmission Measurements

The most common type of measurement to transform is an S11 reflection measurement. However, useful information can be gained about a test device from a transformed S21 transmission measurement. The frequency components pass through the test device and are measured at the B receiver. If there is more than one path through the device, they would appear as various pulses separated in time.

For example, the following transmission measurement shows multiple paths of travel within a Surface Acoustic Wave (SAW) filter. The largest pulse (close to zero time) represents the propagation time of the shortest path through the device. It may not be the largest pulse or represent the desired path. Each subsequent pulse represents another possible path from input to output.



Triple travel is a term used to describe the reflected signal off the output, reflected again off the input, then finally reappearing at the output. This is best seen in a time domain S21 measurement.

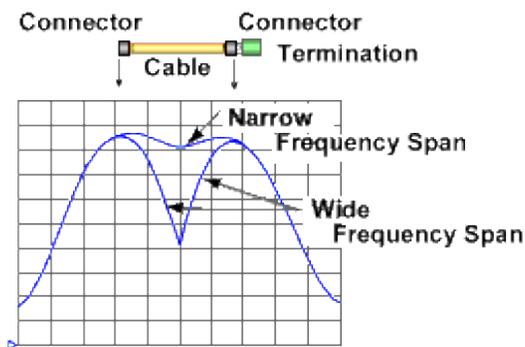
Measurement Response Resolution

In the previous paragraphs, we have seen that using more frequency components causes the assembled waveform to show more detail. This is known as measurement response resolution, which is defined as the ability to distinguish between two closely spaced responses.

Note: Adjusting the [transform time settings](#) improves **display** resolution, but not measurement resolution.

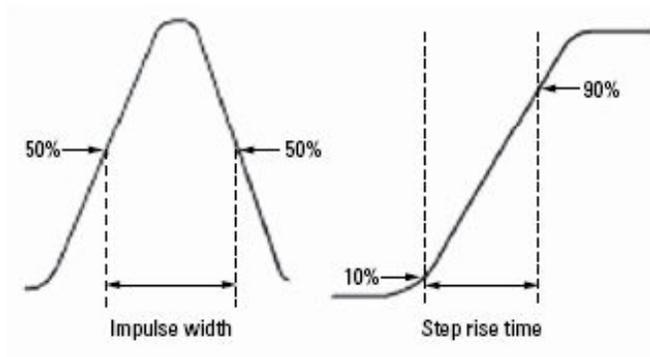
The following graphic shows the effect of both a narrow and wide frequency span on the response resolution. The wider frequency span enables the analyzer to resolve the two connectors into separate,

distinct responses.



Resolution Formula

For responses of equal amplitude, the response resolution is equal to the 50% (-6 dB) points of the impulse width, or the step rise time which is defined as the 10 to 90% points as shown in the following image.



The following table shows the **approximated** relationship between the frequency span and the window selection on response resolution for responses of equal amplitude.

Window	Low-pass step (10% to 90%)	Low-pass impulse (50%)	Bandpass impulse
Minimum	0.45 / f span	0.60 / f span	1.20 / f span
Normal	0.99 / f span	0.98 / f span	1.95 / f span
Maximum	1.48 / f span	1.39 / f span	2.77 / f span

For example, using a 10 GHz wide frequency span and a normal window in Bandpass impulse mode, response resolution (in time) equals:

- Time Res = 1.95 / frequency span
- Time Res = 1.95 / 10 GHz
- Time Res = 195 ps

To calculate the physical separation (in distance) of the responses which can be resolved, multiply this value times the speed of light (c) and the relative velocity (Vf) of propagation in the actual transmission medium. In this case, Vf = 0.66 for polyethylene dielectric.

- Distance Res = 195 ps x c x Vf
- Distance Res = 195 ps x (2.997925 E8 m/s) x .66
- Distance Res = 38 mm

For reflection measurements, because of the 2-way travel time involved, this means that the minimum resolvable separation between discontinuities is half of this value or 19 mm.

Although a wider frequency span causes better measurement resolution, the [measurement range](#) becomes limited. Also, increasing the frequency range can cause a measurement calibration to become invalid. Be sure to adjust the frequency span BEFORE performing a calibration.

Measurement Range and Alias Responses

Measurement range is the length in time in which true time domain responses can be seen. The measurement range should be large enough to see the entire test device response without encountering a repetition (alias) of the response. An alias response can hide a true time domain response.

To increase measurement range in both modes, change either of these settings:

- Increase the number of points
- Decrease the frequency span

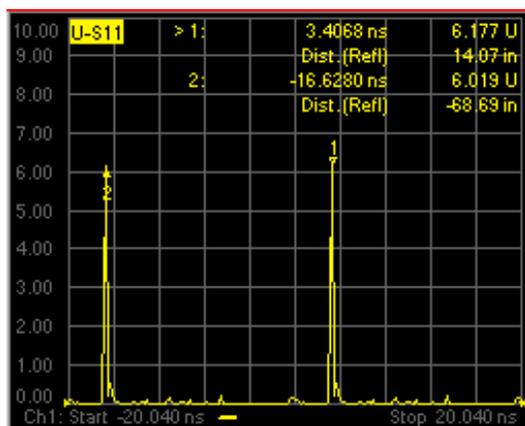
Notes:

- After making these settings, you may need to adjust the [transform time settings](#) to see the new measurement range.
- Decreasing the frequency span degrades [measurement resolution](#).
- Make frequency span and number of points settings BEFORE calibrating.
- Maximum range also depends on loss through the test device. If the returning signal is too small to measure, the range is limited regardless of the frequency span.

Alias Responses

An alias response is not a true device response. An alias response repeats because each time domain waveform has many periods and repeats with time (see [How the PNA Measures in the Time Domain](#)). Alias responses occur at time intervals that are equal to 1/ frequency step size.

The PNA adjusts the [transform time settings](#) so that you should only see one alias free range on either side (positive and negative) of zero time. However, these settings are updated only when one of the toolbar settings are changed.



To determine if a response is true, put a marker on the response and change the frequency span. A true device response will not move in time. An alias response will move.

For example, in the above graphic, the marker 1 response occurs at 14.07 inches. When the frequency span is changed, this response remains at 14.07 inches. The marker 2 response moves.

Range Formula

You can calculate the alias-free measurement range (in meters) of the PNA using the following formula for **TDR** (reflection) measurements:

$$\text{Range (meters)} = (1/ f) \times V_f \times c$$

Where:

- f = frequency step size (frequency span/number of points-1)
- V_f = the velocity factor in the transmission line
- c = speed of light = 2.997925 E8 m/s

For example: For a measurement with 401 points and a span of 2.5 GHz, using a polyethylene cable ($V_f = 0.66$)

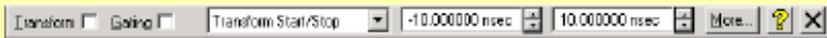
- Range = $(1 / (2.5E9 / 400)) \times 2.997925 \text{ E8 m/s} \times 0.66$
- Range = $6.25E6 \times 2.997925 \text{ E8 m/s} \times 0.66$
- Range = 32 meters

In this example, the range is 32 meters in physical length. To prevent the time domain responses from overlapping or aliasing, the test device must be 32 meters or less in physical length for a transmission measurement.

To calculate the one-way distance for a reflection measurement rather than round-trip distance, simply divide the length by 2. In this case, the alias-free range would be 16 meters.

How to make Time Domain Settings

The following launches the [Time Domain toolbar](#)



On the toolbar, click **More...** to launch the Time Domain dialog box

Using front-panel HARDKEY [softkey] buttons

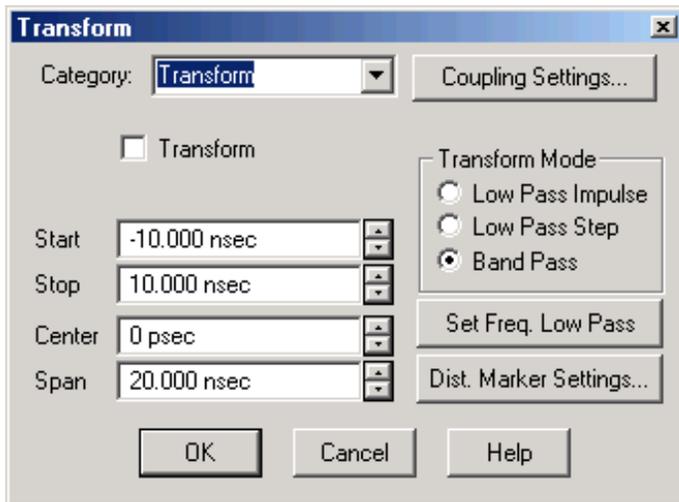
1. Press **ANALYSIS**
2. then **[Transform]**
3. then **[More]**
4. then **[Transform Tool]**

Using a mouse with PNA Menus

1. Click **Marker/Analysis**
2. then **Transform**

Programming Commands

Transform dialog box help



Category Select Transform, Window, or Gating

Transform Turns time domain transform ON and OFF.

Coupling Settings Launches the [Trace Coupling Settings](#) dialog box.

Time Settings

The following settings adjust the **display resolution**, allowing you to zoom IN or OUT on a response. They do NOT adjust [measurement range](#) or [measurement resolution](#).

These settings automatically update (when one of these values are updated) to limit the display to one [alias-free response](#) on either side of zero time.

Start Sets the transform start time that is displayed on the PNA screen.

Note: Zero (0) seconds is always the [measurement reference plane](#). Negative values are useful if moving the reference plane.

Stop Sets the transform stop time that is displayed on the PNA screen.

Center Sets the transform center time that is displayed in the center of the PNA screen.

Span Sets the transform span time that is split on either side of the Center value.

Transform Mode

Transform modes are three variations on how the time domain transform algorithm is applied to the frequency domain measurement. Each method has a unique application.

Mode	Benefit - application	Limitation
Low pass Impulse	Highest resolution. Most useful for seeing small responses in devices that pass low frequencies, such as cables.	In both Low pass modes, frequencies down to DC and negative frequencies are extrapolated. Therefore, the Start frequency is adjusted when you click Set Freq.Low Pass Because this will affect calibration accuracy, be sure to calibrate AFTER completely setting up your time domain measurement.
Low pass Step	Easiest to identify inductive and capacitive discontinuities in devices that pass low frequencies, such as cables.	
Band pass Impulse	Easiest method - can be used with any frequency sweep. Most useful for measuring band limited devices such as filters and DC blocked cables.	Does NOT show capacitive and inductive reactance For the same frequency span and number of points, band pass mode has twice the impulse width, which hides closely spaced responses degrading the response resolution.

The following chart shows how to interpret results from various discontinuity impedances using Low pass Step and either Low pass or Band pass Impulse modes.

IMPEDANCE	STEP RESPONSE	IMPULSE RESPONSE
OPEN	 Unity Reflection	 Unity Reflection
SHORT	 Unity Reflection = 180	 Unity Reflection = 180
RESISTOR $R > Z_0$	 Unity Reflection	 Unity Reflection
RESISTOR $R < Z_0$	 Unity Reflection	 Unity Reflection
INDUCTOR	 Unity Reflection	 Unity Reflection
CAPACITOR	 Unity Reflection	 Unity Reflection

Effect on Measurement Range

Band pass mode - measurement range is inversely proportional to frequency step size.

Low pass mode - measurement range is inversely proportional to the fundamental (start) frequency AFTER clicking Set Freq. Low Pass.

Set Freq. Low Pass USE ONLY IN LOW PASS MODES

Recomputes the start frequency and step frequencies to be harmonics of the start frequency. Start frequency is computed by the following formula: **Low Pass Start Frequency = Stop Frequency / Number of points.**

The computed value must always be greater than or equal to the analyzer's minimum frequency.

Note: The number of points or stop frequency may be changed in order to compute this value.

Distance Marker Settings Launches the [Distance Marker Settings](#) dialog box.

Gating

Perhaps the most beneficial feature of time domain transform is the Gating function. When viewing the time domain response of a device, the gating function can be used to "virtually" remove undesired responses. You can then simultaneously view a frequency domain trace as if the undesired response did not exist.. This allows you to characterize devices without the effects of external devices such as connectors or adapters.

Note: When a discontinuity in a test device reflects energy, that energy will not reach subsequent discontinuities. This can "MASK", or hide, the true response which would have occurred if the previous discontinuity were not present. The PNA Gating feature does NOT compensate for this.

The following measurements images show a practical example how to use and perform gating. The test device is a 10inch cable, then a 6 dB attenuator, terminated with a short. The following four discontinuities are evident in window 2, from left to right:

1. A discontinuity in the test system cable which appeared after calibration. It is identified by marker 2 at -10.74 inches (behind the reference plane).
2. A discontinuity in the 10 inch device cable shortly after the reference plane.
3. The largest discontinuity is the attenuator and short shown by marker 1 at -12.67 dB (6 dB loss in both forward and reverse direction).
4. The last discontinuity is a re-reflection from the device cable.

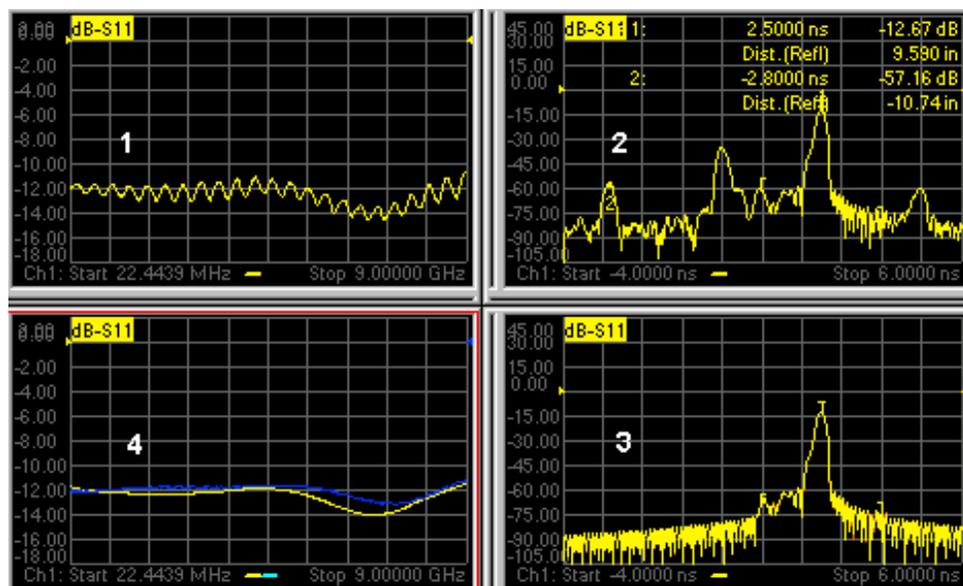
We will gate IN the attenuator response. All other responses will be gated OUT.

Window 1. Create original S11 frequency domain trace. Shows ripple from all of the reflections.

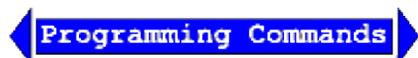
Window 2. Create a new S11 trace - same channel; new window. Turn Transform ON.

Window 3. On the transformed trace, turn gating ON. Center the gate on the large discontinuity (2.500ns). Adjust gate span to completely cover the discontinuity. Select Bandpass gating type.

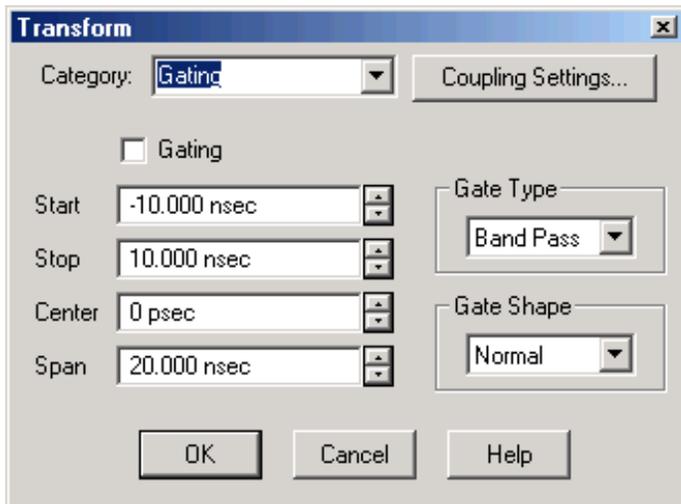
Window 4. On the original frequency measurement, turn Gating ON (Transform remains OFF). View the measurement without the effects of the two unwanted discontinuities. The blue trace is a measurement of the 6 dB attenuator with the unwanted discontinuities PHYSICALLY removed. The difference between the two traces in window 4 is the effect of "[masking](#)".



[Learn how to launch the Transform dialog box](#)



[Transform Gating dialog box help](#)



Gating Turns Gating ON and OFF.

Coupling Settings Launches the [Setup Trace Coupling](#) dialog box.

Start Specifies the start time for the gate.

Stop Specifies the stop time for the gate.

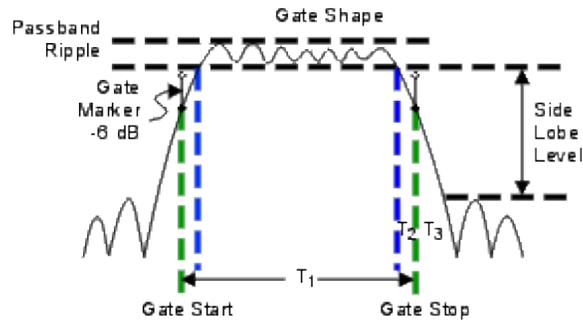
Center Specifies the value at the center of the area that is affected by the gating function. This value can be anywhere in the analyzer range.

Span Specifies the range to either side of the center value of area that is affected by the gating function.

Gate Type Defines the type of filtering that will be performed for the gating function. The gate start and stop flags on the display point toward the part of the trace you want to keep.

- **Bandpass** - KEEPS the responses within the gate span.
- **Notch** - REMOVES the responses with the gate span.

Gate Shape Defines the filter characteristics of the gate function. Choose from Minimum, Normal, Wide, Maximum



Gate Shape	Passband Ripple	Sidelobe Levels	Cutoff Time	Minimum Gate Span
Minimum	± 0.1 dB	-48 dB	1.4/Freq Span	2.8/Freq Span
Normal	± 0.1 dB	-68 dB	2.8/Freq Span	5.6/Freq Span
Wide	± 0.1 dB	-57 dB	4.4/Freq Span	8.8/Freq Span
Maximum	± 0.01 dB	-70 dB	12.7/Freq Span	25.4/Freq Span

Cutoff time -- is the time between the stop time (-6 dB on the filter skirt) and the peak of the first sidelobe. The diagram below shows the overall gate shape and lists the characteristics for each gate shape.

- T₁ is the gate span, which is equal to the stop time minus the start time.
- T₂ is the time between the edge of the passband and the 6 dB point, representing the cutoff rate of the filter.
- T₃ is the time between the 6 dB point and the edge of the gate stopband.
- For all filter shapes T₂ is equal to T₃, and the filter is the same on both sides of the center time.

Minimum gate span -- is twice the cutoff time. Each gate shape has a minimum recommended gate span for proper operation. This is a consequence of the finite cutoff rate of the gate. If you specify a gate span that is smaller than the minimum span, the response will show the following effects:

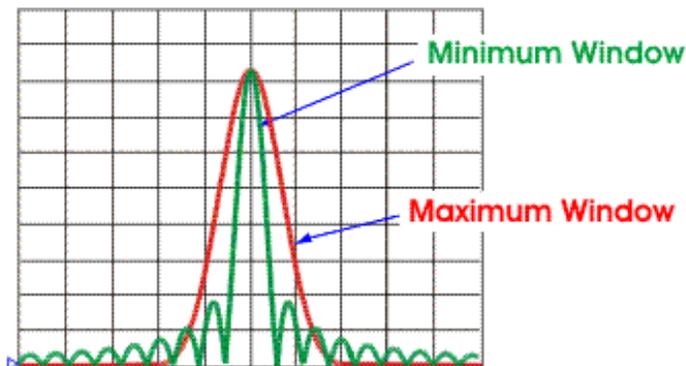
- distorted gate shape that has no passband
- distorted shape
- incorrect indications of start and stop times
- may have increased sidelobe levels

Window Settings

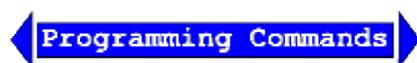
There are abrupt transitions in a frequency domain measurement at the start and stop frequencies,

causing overshoot and ringing in a time domain response. The window feature is helpful in lessening the abruptness of the frequency domain transitions. This causes you to make a tradeoff in the time domain response. Choose between the following:

- **Minimum Window = Better Response Resolution** - the ability resolve between two closely spaced responses.
- **Maximum Window = Dynamic Range** - the ability to measure low-level responses.



[Learn how to launch the Transform dialog box](#)



Transform - Window dialog box help

Category: Window

Minimum Maximum

Kaiser Beta 6.000

Impulse Width 43.367 psec

Coupling Settings Launches the [Setup Trace Coupling](#) dialog box.

The window settings balance response resolution versus dynamic range.

- Minimum Window = Best Response Resolution
- Maximum Window = Best Dynamic Range

The following three methods all the set window size. For best results, view the time domain response while making these settings.

- **Minimum - Maximum** Move the slider with a mouse to change the window size
- **Kaiser Beta** Changes window size using a Kaiser Beta value
- **Impulse Width** Changes window size using an Impulse Width value

Learn more about [Windowing](#) (top)

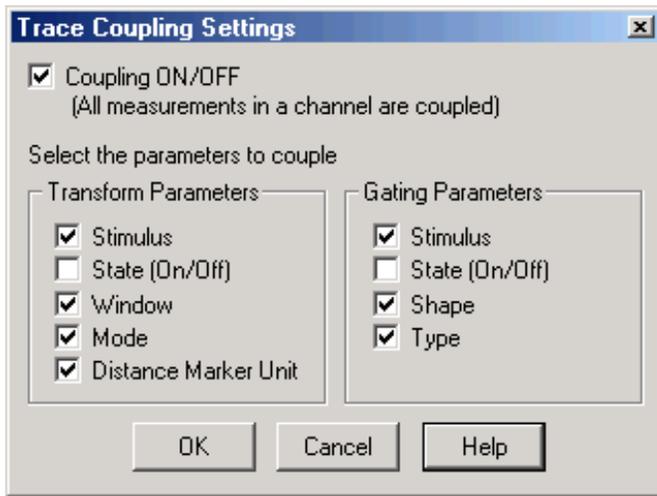
How to make Trace Coupling Settings

You can launch the **Trace Coupling Settings** dialog box from any of the following dialog boxes:

- [Transform](#)
- [Gating](#)
- [Window](#)

◀ Programming Commands ▶

Trace Coupling Settings dialog box help



Trace coupling allows you to change time domain parameters on a measurement, and have the same changes occur for all other measurements in the channel.

For example:

If you are simultaneously viewing a frequency domain measurement and time domain measurement,

and **Coupling** is enabled in this dialog box,

and **ALL Gating Parameters** are checked in this dialog box,

and on the time domain measurement you change the **Gate Span** parameter,

Then the frequency domain measurement will automatically change to reflect the time domain gated span.

Note: Trace coupling applies **ONLY** to the Y-axis scale/reference settings. There are no changes to your data as a result of trace coupling.

Coupling ON/OFF Check to enable coupling. All of the measurements in the active channel are coupled.

The following parameters are available for coupling:

Transform Parameters

Stimulus Start, Stop, Center, and Span TIME settings.

State (On/Off) Transform ON and OFF

Window Kaiser Beta / Impulse Width

Mode Low Pass Impulse, Low Pass Step, Band Pass

Gating Parameters

Stimulus Start, Stop, Center, and Span TIME settings.

State (On/Off) Gating ON and OFF

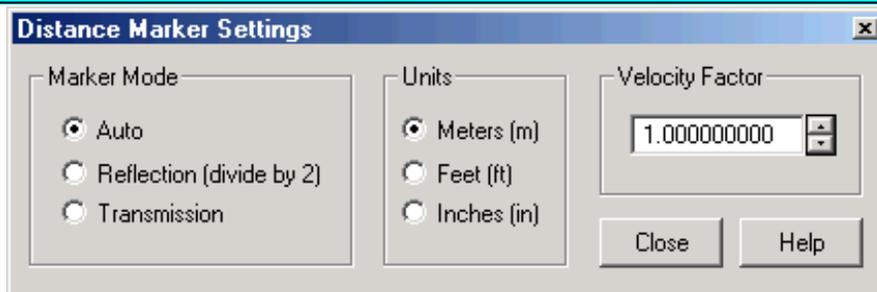
Shape Minimum, Normal, Wide, and Maximum

Type Bandpass and Notch

To launch the Distance marker dialog box, click **Dist. Marker Settings** on the [Transform](#) dialog box.



Distance Marker Settings dialog box help



When markers are present on a time domain measurement, distance is automatically displayed on the marker readout, [marker table](#), and [print copy](#). To learn how to create markers on your measurement see [marker settings](#).

You can read out impedance versus time by creating a marker on a Time Domain trace, then changing the marker format to R+jX. [Learn how](#).

This dialog box allows you to customize the time domain distance marker readings.

These settings affect the display of ALL markers for only the ACTIVE measurement (unless **Distance Marker Unit** is coupled on the [Trace Coupling dialog box](#)).

Marker Mode Specifies the measurement type in order to determine the correct marker distance.

- Select **Auto** for [S-Parameter](#) measurements.
- Select **Reflection** or **Transmission** for [arbitrary ratio](#) or [unratioed](#) measurements.

Auto If the active measurement is an S-Parameter, automatically chooses reflection or transmission. If the active measurement is a non S-Parameter, reflection is chosen.

Reflection Displays the distance from the source to the receiver and back divided by two (to compensate for the return trip.)

Transmission Displays the distance from the source to the receiver.

Units Specifies the unit of measure for the display of marker distance values.

Velocity Factor Specifies the velocity factor that applies to the medium of the device that was inserted after the measurement calibration. The value for a polyethylene dielectric cable is 0.66 and 0.7 for PTFE dielectric. 1.0 corresponds to the speed of light in a vacuum. This is useful in Time Domain for accurate display of time and distance markers.

This setting can also be made from the [Electrical Delay](#) and [Port Extensions](#) dialog boxes.

Last Modified:

3-Apr-2013	Added marker impedance note
18-Mar-2013	Update to trace couple settings
3-Sep-2008	Removed legacy content
11-Sep-2007	Edits to resolution and range formulas

Wideband Pulsed Application

Note: The PNA-X with Opt 025 and Opt 008 has an integrated [Pulse Measurement Setup](#) that makes this application no longer necessary on the PNA-X.

The Wideband Pulsed Application configures the PNA-X internal pulse generators and modulators for measuring pulsed S-parameters using the wideband mode detection technique.

The Wideband Pulse Application is designed to be used with the PNA-X with Opts 021, 022, and 025.

Note: Wideband Pulse application is NOT supported on the E836x and PNA-L models.

See Also

- To learn more about wideband detection, see [Application Note 1408-12](#).
- See a Visual Basic example: [Create a Wideband Pulsed Measurement using the PNA-X](#)
- Learn about the [Narrowband Pulsed Application](#).

Download and Install the Wideband Pulsed Application

This application is installed and run as a macro on the PNA-X. [Learn more about macros](#).

1. Go to <http://na.tm.agilent.com/pna/apps/applications.htm>
2. Click the download link
3. Save the downloaded file to the PNA hard drive
4. Double-click the downloaded file to install the Wideband Pulsed Application on the PNA.
5. The application is saved on the PNA at C:/Program Files/Agilent/Network Analyzer/Applications/WB Pulse/Wideband_pulse.exe
6. Configure the macro. [Learn how](#).

To learn more about Wideband pulsed application, click Help in the application.

Last Modified:

17-Mar-2010	Added PNA-X note
22-Jun-2007	MX New topic

Troubleshooting the PNA

By running a few checks, you can identify if the analyzer is at fault. Before calling Agilent Technologies or returning the instrument for service, please make the following checks.

- [Check the Basics](#)
- [PNA Application Terminates Unexpectedly](#)
- [Check Error Terms](#)
- [Check the Service Guide](#)

Other Support Topics

Check the Basics

A problem can often be solved by repeating the procedure you were following when the problem occurred. Before calling Agilent Technologies or returning the instrument for service, please make the following checks:

Note: Problems with the PNA application (slow or terminates unexpectedly) can be caused by a faulty Hard Disk Drive (HDD). For more information, see [Preventing PNA Hard Drive Problems](#) and [The PNA HDD Recovery Process](#).

1. Is there power at the power socket? Is the instrument plugged in?
2. Is the instrument turned on? Check to see if the front panel line switch and at least one of the LED rings around the test ports glows green. This indicates the power supply is on.
3. If you are experiencing difficulty with the front-panel keypad or peripherals, the USB bus may be overloaded. Remove the USB devices, restart the PNA, and reconnect the USB devices. See [Power-up](#).
4. If other equipment, cables, and connectors are being used with the instrument, make sure they are connected properly and operating correctly.
5. Review the procedure for the measurement being performed when the problem appeared. Are all the settings correct?
6. If the instrument is not functioning as expected, return the unit to a known state by pressing the **Preset** key.
7. Is the measurement being performed, and the results that are expected, within the [specifications](#) and capabilities of the instrument?
8. If the problem is thought to be due to firmware, check to see if the instrument has the [latest firmware](#) before starting the troubleshooting procedure.

9. Check that the measurement calibration is valid. See [Accurate Measurement Calibrations](#) for more information.
10. If the necessary test equipment is available, perform the operator's check and system verification in Chapter 2 of the PNA Service Guide, "System Tests, Verifications, and Adjustments," . You can download a copy of the Service Guide from our Web site: <http://www.agilent.com>.
11. **Phase lock lost message** - This usually occurs when there is not enough source power to phase lock the PNA. It can occur during an errant [FCA setup](#) or [Source Power Calibration](#). It can also occur if one of the front panel reference channel loops is not connected. Otherwise, this indicates a hardware problem.

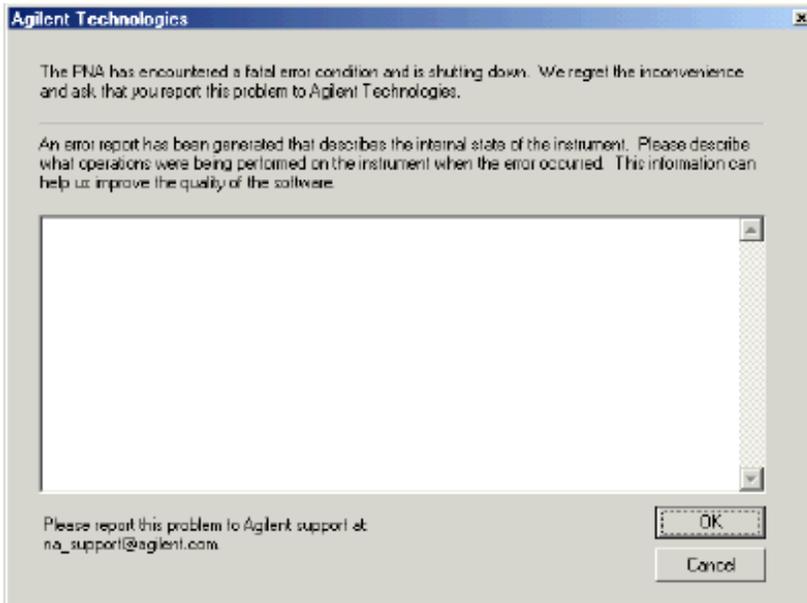
PNA Application Terminates Unexpectedly

If an unexpected and irrecoverable error occurs, Agilent would like to know about it. The PNA attempts to save pertinent information about the state of the system. **The PNA does NOT send this information to Agilent.**

We respect the privacy of our customers. However, access to information that helps us improve the PNA is a benefit to both Agilent and you. Please take the time to contact us or email the saved information to na_support@agilent.com.

The following procedure shows how to do this:

1. A message box immediately appears on the screen containing the location of a directory. Please record this message. If you miss the message, you can find the directory location using the Windows Event Log: On the PNA, click Start, Settings, Control Panel, Administrative Tools, Event Viewer. Double-click the top line (most recent event). The location of the directory is seen in the Description.
2. A dialog box may appear on the screen (shown below) allowing you to add comments to help us replicate the crash.
3. Find the directory (described in Step 1) which contains the following files:
 - 835x.dmp which is the 835x.exe capturing the context in which the program crashed.
 - 835x.xml which reports some very basic information (exception code, OS version, and the list of modules loaded at the time of the crash and their respective version numbers).
 - 835xCrashLog.txt: The text file with your comments (described in Step 2), if submitted.
4. If your PNA is not connected to LAN or is not configured to send email, copy the files to a PC. Then, please email the files to na_support@agilent.com



Check Error Terms

If you print the error terms at set intervals (weekly, monthly, and so forth), you can compare current error terms to these records. A stable, repeatable system should generate repeatable error terms over long time intervals, for example, six months. If a subtle failure or mild performance problem is suspected, the magnitude of the error terms should be compared against values generated previously with the same instrument and calibration kit. See the [procedure for monitoring error terms](#).

- A long-term trend often reflects drift, connector and cable wear, or gradual degradation, indicating the need for further investigation and preventative maintenance. Yet, the system may still conform to specifications. The cure is often as simple as cleaning and gaging connectors or inspecting cables.
- A sudden shift in error terms reflects a sudden shift in systematic errors, and may indicate the need for further troubleshooting.

Consider the following while troubleshooting:

- All parts of the system, including cables and calibration devices, can contribute to systematic errors and impact the error terms.
- Connectors must be clean and gauged, and within specification for error term analysis to be meaningful. See the Chapter 2 in the PNA Service Guide for information on cleaning and gaging connectors.
 - Avoid unnecessary bending and flexing of the cables following measurement calibration, thus minimizing cable instability errors.
 - Use good connection techniques during the measurement calibration. The connector interface must be repeatable. See the PNA Service Guide for information on connection techniques.

- It is often worthwhile to perform the procedure twice (using two distinct measurement calibrations) to establish the degree of repeatability. If the results do not seem repeatable, check all connectors and cables.
- Use error-term analysis to troubleshoot minor, subtle performance problems. See Chapter 3, "Troubleshooting," in the PNA Service Guide if a blatant failure or gross measurement error is evident.

Check the Service Guide

Check the PNA Service Guide for specific troubleshooting procedures to help identify problems. You can download a copy of the Service Guide from our Web site: <http://www.agilent.com>.

Last modified:

10/16/06 Added phase lock lost

PNA Error Messages

- [500 - 750 Calibrate](#)
- [770 - 1000 Hardware](#)
- [1000 - 1200 Measure](#)
- [1281 - 1535 Parser](#)
- [1536 - 1650 Display](#)
- [1700 - 2000 Channel](#)
- [2048 - 2200 General](#)
- [Standard SCPI Errors](#)

Note: The **EventID**'s listed below are provided for COM programming. For more information, see [Working with PNA Events](#)

For more information on PNA error messages ([see Error Messages](#)).

Memory Overflow Error

Memory overflow. Trigger state set to Hold. Lower the IF bandwidth, or increase dwell or sweep time.

Severity: Informational

Further explanation: The measurement that you are currently making requires that data be stored faster than it can be processed. Very few customers will experience this situation.

Suggestions: To limit the amount of data to be stored, try lowering the IF Bandwidth, slow the sweep time, increase the dwell time, or limit the number of data points. There are many other settings that can be adjusted to solve this problem.

EventID:

Cal Errors

Message: 512

"A secondary parameter (power, IFBW, sweep time, step mode) of the calibrated state has changed."

Severity: Informational

Further explanation: The calibration is questionable when any of these secondary parameters change after the calibration is performed.

Suggestions: If you require an accurate measurement with the new settings, repeat the calibration.

EventID: 68020200 (hex)

Message: 513

"Calibration cannot be completed until you have measured all the necessary standards for your selected Cal Type."

Severity: Informational

Further explanation: You probably received this message because you attempted to turn correction on without first measuring all of the calibration standards

Suggestions: Finish measuring the cal standards

EventID: 68020201 (hex)

Message: 514

"Calibration set has been recalled using a file previously saved on an analyzer that had a different hardware configuration."

Severity: Informational

Further explanation:

Suggestions:

EventID: 68020202 (hex)

Message: 515

"Calibration is required before correction can be turned on. Channel number is <x>, Measurement is <x>."

Severity: Informational

Further explanation: There are no error correction terms to apply for the specified channel and measurement.

Suggestions: Perform or recall a calibration

EventID: 68020203 (hex)

Message: 516

"Critical parameters in your current instrument state do not match the parameters for the calibration set, therefore correction has been turned off. The critical instrument state parameters are sweep type, start frequency, frequency span, and number of points."

Severity: Informational

Further explanation: None

Suggestions: You can either recalibrate using the new settings or change back to the original setting that was used when the calibration was performed.

EventID: 68020204 (hex)

Message: 517

"Interpolation is turned off and you have changed the stimulus settings of the original calibration, so correction has been turned off."

Severity: Informational

Further explanation: The most accurate calibration is maintained only when the original stimulus settings are used.

Suggestions: If reduced accuracy is OK, set interpolation ON to allow stimulus setting changes.

EventID: 68020205 (hex)

Message: 518

"Interpolation is turned off and you have selected correction ON. Correction has been restored with the previous stimulus settings."

Severity: Informational

Further explanation: None

Suggestions: None

EventID: 68020206 (hex)

Message: 519

"Stimulus settings for your current instrument state exceeded the parameters of the original calibration, so correction has been turned off."

Severity: Informational

Further explanation: Correction data outside the stimulus settings does not exist.

Suggestions: Perform a broadband calibration, with increased numbers of points with interpolation ON, to maintain calibration over the widest possible stimulus frequency settings.

EventID: 68020207(hex)

Message: 520

"Cal Type is set to NONE for Channel <x>, Measurement <x>; please select Calibration menu or press Cal hard key."

Severity: Informational

Further explanation: A cal operation can not proceed until a calibration exists or the cal type is selected. This error can occur if the calibration can not be found. Also this error can happen if a calibration type is not specified before attempting to programmatically execute cal acquisitions.

Suggestions To find a calibration, select a Cal Set that contains the calibration needed for the current measurements. OR specify the cal type before beginning a calibration procedure.

EventID: 68020208 (hex)

Message: 521

"The measurement you set up does not have a corresponding calibration type, so correction has been turned off or is not permitted."

Severity: Informational

Further explanation: The calibration for the channel may apply only to certain S-Parameters. For example, a 1-Port calibration for S11 can not be applied to a 1-Port calibration applied to S22.

Suggestions: Select a calibration type, such as full 2-Port cal, that can be applied to all the measurements to be selected.

EventID: 68020209 (hex)

Message: 522

"The calibration type you selected cannot be set up."

Severity: Informational

Further explanation: "Please use the SCPI command [ROUTe:PATH:DEFine:PORT](#) <num>,<num> for full 2 port type port assignment."

Suggestions:

EventID: 6802020A (hex)

Message: 523

"The calibration path you selected cannot be set up because it is not valid for the current measurement."

Severity: Informational

Further explanation: "Please use the SCPI command [ROUTE:PATH:DEFine:PORT](#) <num>,<num> for full 2 port type port assignment related to your current measurement."

Suggestions:

EventID: 6802020B (hex)

Message: 524

"The source power calibration is complete."

Severity: Informational

Further explanation:

Suggestions:

EventID: 6802020C (hex)

Message: 525

"You have specified more than 7 standards for one or more calibration classes."

Severity: Informational

Further explanation: These have been truncated to 7 selections.

EventID: 6802020D (hex)

Message: 526

"No user calibration found for this channel."

Severity: Informational

Further explanation: A cal operation can not proceed until a calibration exists.

Suggestions: To find a calibration, you can select a Cal Set that contains the calibration needed for the

current measurement.

EventID: 6802020E (hex)

Message: 527

"You do not need to acquire this standard for this calibration type."

Severity: Informational

Further explanation: This error can happen as a result of PROGRAMMATICALLY requesting the measurement of an un-needed calibration standard during a calibration procedure.

Suggestions: Check the specified cal type or eliminate the request for the measurement of the standard.

EventID: 6802020F (hex)

Message: 528

"Could not configure the Electronic Calibration system. Check to see if the module is plugged into the proper connector."

Severity: Informational

Further explanation: During an ECal operation, communication could not be established with the ECal module. The calibration will not be initiated until the presence of the ECal module is verified.

Suggestions: Verify the USB cable is connected properly. Disconnect and re-connect the cable to ensure the analyzer recognizes the module.

EventID: 68020210 (hex)

Message: 529

"DATA OUT OF RANGE: Design Limits Exceeded"

Severity: Error

Further explanation:

Suggestions:

EventID: E8020211(hex)

Message: 530

"EXECUTION ERROR: Could not open ECal module memory backup file"

Severity: Error

Further explanation:

Suggestions:

EventID: E8020212 (hex)

Message: 531

"EXECUTION ERROR: Access to ECal module memory backup file was denied"

Severity: Error

Further explanation:

Suggestions:

EventID: E8020213 (hex)

Message: 532

"EXECUTION ERROR: Failure in writing to ECal module memory backup file"

Severity: Error

Further explanation:

Suggestions:

EventID: E8020214 (hex)

Message: 533

"EXECUTION ERROR: Failure in reading from ECal module memory backup file"

Severity: Error

Further explanation:

Suggestions:

EventID: E8020215 (hex)

Message: 534

"EXECUTION ERROR: Array index out of range"

Severity: Error

Further explanation:

Suggestions:

EventID: E8020216 (hex)

Message: 535

"EXECUTION ERROR: Arrays wrong rank"

Severity: Error

Further explanation:

Suggestions:

EventID: E8020217 (hex)

Message: 536

"EXECUTION ERROR: CPU"

Severity: Error

Further explanation:

Suggestions:

EventID: E8020218 (hex)

Message: 537

"EXECUTION ERROR: Cannot ERASE module"

Severity: Error

Further explanation:

Suggestions:

EventID: E8020219 (hex)

Message: 538

"EXECUTION ERROR: Cannot WRITE module"

Severity: Error

Further explanation:

Suggestions:

EventID: E802021A (hex)

Message: 539

"EXECUTION ERROR: Entry Not Found"

Severity: Error

Further explanation:

Suggestions:

EventID: E802021B (hex)

Message: 540

"EXECUTION ERROR: Invalid command while system is busy"

Severity: Error

Further explanation:

Suggestions:

EventID: E802021C (hex)

Message: 541

"Electronic Cal: Unable to orient ECal module. Please ensure the module is connected to the necessary measurement ports."

Severity: Error

Further explanation: There is no RF connection to the ECal module during a calibration step. An ECal orientation measurement has been attempted but the signal was not found.

Suggestions: Connect the ECal module RF connections to ports specified for the calibration step. The ECal module typically requires at least -18dBm for measurements. If your measurement requires the power level to be less than that, clear the **Do orientation** checkbox to bypass the automatic detection step.

EventID: E802021D (hex)

Message: 542

"EXECUTION ERROR: NO SPACE for NEW CAL, DELETE A CAL"

Severity: Error

Further explanation:

Suggestions:

EventID: E802021E (hex)

Message: 543

"EXECUTION ERROR: No More Room"

Severity: Error

Further explanation:

Suggestions:

EventID: E802021F (hex)

Message: 544

"EXECUTION ERROR: Other array error"

Severity: Error

Further explanation:

Suggestions:

EventID: E8020220 (hex)

Message: 545

"EXECUTION ERROR: Ranks not equal"

Severity: Error

Further explanation:

Suggestions:

EventID: E8020221 (hex)

Message: 546

"EXECUTION ERROR: Too few CONSTANT ranks"

Severity: Error

EventID: E8020222 (hex)

Message: 547

"EXECUTION ERROR: Too few VARYing ranks"

Severity: Error

EventID: E8020223 (hex)

Message: 548

"EXECUTION ERROR: Unknown error"

Severity: Error

EventID: E8020224 (hex)

Message: 549

"EXECUTION ERROR: ecaldrv.dll bug or invalid module #"

Severity: Error

EventID: E8020225 (hex)

Message: 550

"EXECUTION ERROR: unexpected error code from ecal driver"

Severity: Error

EventID: E8020226 (hex)

Message: 551

"EXECUTION ERROR: unexpected internal driver error"

Severity: Error

EventID: E8020227 (hex)

Message: 552

"HARDWARE ERROR: Can't access ECal Interface Module"

Severity: Error

EventID: E8020228 (hex)

Message: 553

"HARDWARE ERROR: Can't release LPT port, reboot"

Severity: Error

EventID: E8020229 (hex)

Message: 554

"HARDWARE ERROR: VNA Error"

Severity: Error

EventID: E802022A (hex)

Message: 555

"HARDWARE ERROR: not enough data read from ECal module"

Severity: Error

EventID: E802022B (hex)

Message: 556

"OPERATION ABORTED BY HOST COMPUTER"

Severity: Error

EventID: E802022C (hex)

Message: 557

"OPERATION ABORTED BY USER"

Severity: Error

EventID: E802022D (hex)

Message: 558

"OUT OF MEMORY"

Severity: Error

EventID: E802022E (hex)

Message: 559

"QUERY INTERRUPTED:Message(s Abandoned"

Severity: Error

EventID: E802022F (hex)

Message: 560

"QUERY UNTERMINATED: INCOMPLETE PROGRAM Message"

Severity: Error

Further explanation:

Suggestions:

EventID: E8020230 (hex)

Message: 561

"QUERY UNTERMINATED: NOTHING TO SAY"

Severity: Error

Further explanation:

Suggestions:

EventID: E8020231 (hex)

Message: 562

"QUEUE OVERFLOW"

Severity: Error

EventID: E8020232 (hex)

Message: 563

"SETTINGS CONFLICT: ADDITIONAL STANDARDS ARE NEEDED"

Severity: Error

EventID: E8020233 (hex)

Message: 564

"SETTINGS CONFLICT: Adapter Cal is NOT possible"

Severity: Error

EventID: E8020234 (hex)

Message: 565

"SETTINGS CONFLICT: COMMAND OUT OF SEQUENCE"

Severity: Error

EventID: E8020235 (hex)

Message: 566

"SETTINGS CONFLICT: Cal STOPPED - VNA SETUP CHANGED"

Severity: Error

EventID: E8020236 (hex)

Message: 567

"SETTINGS CONFLICT: Calibration is NOT in progress"

Severity: Error

EventID: E8020237 (hex)

Message: 568

"SETTINGS CONFLICT: Can't find specified GPIB board"

Severity: Error

EventID: E8020238 (hex)

Message: 569

"SETTINGS CONFLICT: Can't find/load gpib32.dll"

Severity: Error

EventID: E8020239 (hex)

Message: 570

"SETTINGS CONFLICT: Can't find/load sicl32.dll"

Severity: Error

EventID: E802023A (hex)

Message: 571

"SETTINGS CONFLICT: Can't initialize VNA (bad address?"

Severity: Error

EventID: E802023B (hex)

Message: 572

"SETTINGS CONFLICT: Can't load LPT port driver or USB driver DLL"

Severity: Error

EventID: E802023C (hex)

Message: 573

"SETTINGS CONFLICT: Invalid Calibration Sweep Mode."

Severity: Error

EventID: E802023D (hex)

Message: 574

"SETTINGS CONFLICT: Invalid Calibration Type"

Severity: Error

EventID: E802023E (hex)

Message: 575

"SETTINGS CONFLICT: Invalid Calibration"

Severity: Error

EventID: E802023F (hex)

Message: 576

"SETTINGS CONFLICT: Invalid GPIB board number specified"

Severity: Error

EventID: E8020240 (hex)

Message: 577

"SETTINGS CONFLICT: Invalid GPIB board type specified"

Severity: Error

EventID: E8020241 (hex)

Message: 578

"SETTINGS CONFLICT: Invalid Module Status"

Severity: Error

EventID: E8020242 (hex)

Message: 579

"SETTINGS CONFLICT: Invalid States"

Severity: Error

EventID: E8020243 (hex)

Message: 580

"SETTINGS CONFLICT: LPT port must be between 1 and 4"

Severity: Error

EventID: E8020244 (hex)

Message: 581

"Could not configure the Electronic Calibration system. Check to see if the module is properly connected."

Severity: Error

EventID: E8020245 (hex)

Message: 582

"SETTINGS CONFLICT: Specified LPT port does not exist"

Severity: Error

EventID: E8020246 (hex)

Message: 583

"SETTINGS CONFLICT: Use frequency domain for cal"

Severity: Error

EventID: E8020247 (hex)

Message: 584

"SETTINGS CONFLICT: Use step sweep type for cal."

Severity: Error

EventID: E8020248 (hex)

Message: 585

"SETTINGS CONFLICT: VNA address must be between 0 and 30"

Severity: Error

EventID: E8020249 (hex)

Message: 586

"SETTINGS CONFLICT: Wrong LPT port driver or USB driver DLL"

Severity: Error

EventID: E802024A (hex)

Message: 587

"SYNTAX ERROR: ECAL:DELAY command must have 2 numbers"

Severity: Error

EventID: E802024B (hex)

Message: 588

"SYNTAX ERROR: INCORRECT SYNTAX"

Severity: Error

EventID: E802024C (hex)

Message: 589

"SYNTAX ERROR: UNKNOWN COMMAND"

Severity: Error

EventID: E802024D (hex)

Message: 590

"Wrong port of module in RF path"

Severity: Error

EventID: E802024E (hex)

Message: 591

"User characterization not found in module"

Severity: Error

EventID: E802024F (hex)

Message: 592

Severity: Informational

"No source power calibration found for the channel and source port of the current measurement."

Further explanation: You tried to turn on source power cal but there is no source power cal data.

Suggestions: Perform a source power calibration

EventID: 68020250 (hex)

Message: 593

Severity: Informational

"A source power calibration sweep was not performed, so there is no correction for the channel and

source port of the current measurement."

Further explanation: You tried to turn on source power cal but there is incomplete source cal data.

Suggestions: Perform a complete source power calibration

EventID: 68020251 (hex)

Message: 594

Severity: Informational

"A new trace could not be added to the active window for viewing the source power cal sweep, because it would have exceeded the limit on number of traces/window. Please remove a trace from the window before proceeding with source power cal."

Further explanation: The source power cal attempts to add a data trace to the active window. The active window already contains four traces.

Suggestions: Make the active window contain less than four traces.

EventID: 68020252 (hex)

Message: 595

Severity: Informational

"A new measurement could not be added for performing the source power cal sweep, because the limit on number of measurements has been reached. Please remove a measurement before proceeding with source power cal."

Further explanation: The source power cal attempts to add a measurement. The PNA already has the maximum number of measurements.

Suggestions: Delete a measurement.

EventID: 68020253 (hex)

Message: 596

Severity: Informational

"The calibration power value associated with the source power calibration of Port %1 on Channel %2 was changed with the calibration on. The calibration was not turned off, but the power value might no longer represent the calibration."

Further explanation: The source power cal accuracy is questionable.

Suggestions: If high accuracy is required, perform another source power calibration.

EventID: 68020254 (hex)

Message: 597

Severity: Informational

- Message that is passed from the power meter driver for a source power calibration. -

Further explanation: This error is generated by the power meter driver and passed through the PNA.

EventID: 68020255 (hex)

Message: 598

"During the acquisition of the sliding load standard, the slide was not properly moved to perform a circle fit. The standard's raw impedance was used to determine the directivity for one or more points."

Severity: Informational

Further Explanation: To accurately characterize the standard, the sliding load must be move sufficiently to ensure enough samples around the complex circle or Smith Chart. Under-sampling will cause an inaccurate result.

Suggestions: For best results when using a sliding load, be sure to use multiple slide positions that cover the full range of movement from front to back of the slot.

EventID: 68020256 (hex)

Message: 599

"This feature requires an unused channel, but could not find one. Please free up a channel and try again."

Severity: Informational

Further Explanation: You attempted to view an item within a calset. However, the calset viewer requires that the result be displayed in a channel that is not currently in use. All the channels are currently used. The view can not display the requested item.

Suggestions: You must delete at least one channel that is currently in use.

EventID: 68020257 (hex)

Message: 600

"Interpolation of the original calibration is not allowed since it was performed using Segment Sweep. Correction has been turned off."

Severity: Informational

EventID: 68020258 (hex)

Message: 601

"Cal preferences saved. Cal preference settings can be changed from the 'Cal Preferences' drop down Cal menu."

Severity: Informational

Further explanation: [See Save Preference](#)

EventID: 68020259 (hex)

Message: 608

"CalType not set."

Severity: Error

Further explanation: A cal operation can not proceed until a calibration exists or the proper cal type is selected.

Suggestions: This error can happen if the calibration can't be found. To find a calibration, you can select a Cal Set that contains the calibration needed for the current measurements. This error can also happen if a calibration type is not specified before attempting to programmatically execute cal acquisitions. Specify the cal type before beginning a calibration procedure.

EventID: E8020260 (hex)

Message: 609

"The Calibration feature requested is not implemented."

Further explanation: The specified cal type can be one of many choices. For example, response calibrations require single standards, 1-Port calibrations require 3 standards, and 2-Port calibrations require up to 12 standards.

Suggestions: Be sure to measure only the standards needed for the specified cal type.

EventID: E8020261 (hex)

Message: 610

"The Calibration Class Acquisition requested is not valid for the selected Calibration Type. Please select a different acquisition or a different Calibration Type."

EventID: E8020262 (hex)

Message: 611

"The Calibration Standard data required for the selected caltype was not found."

Severity: Error

Further explanation: An unsuccessful attempt was made to retrieve a specified standard from the raw measurement buffer. The buffer should contain the raw measurements of cal standards stored during a calibration procedure.

Suggestions: Be sure the requested standard is required for the current cal type. Not all standards are needed for all cal types.

EventID: E8020263 (hex)

Message: 612

" The Error Term data required for the selected caltype was not found."

Severity: Error

Further explanation: An unsuccessful attempt was made to retrieve a specified error term from the error correction buffer. The buffer should contain the error correction arrays for the current calibration.

Suggestions: Be sure the requested error term is required for the current cal type. Not all error terms are needed for all cal types.

EventID: E8020264 (hex)

Message: 613

The Calibration data set was not found.

Severity: Error

Further explanation: An unsuccessful attempt to access a cal set has been made. This may indicate a calset has been deleted or has been corrupted.

Suggestions: Try again or select another cal set. If the cal set appears in the cal set list, it may need to be deleted.

EventID: E8020265 (hex)

Message: 614

"The specified measurement does not have a calibration valid for Confidence Check. Please select a different measurement, or recall or perform a different Calibration Type."

Severity: Error

Further explanation: The measurement choice is prevented so that calibration will not be turned off. Not all cal types support all measurements. For example, an 1-Port cal on S11 can not be used to calibrate an S12 measurement. When a measurement is selected that does not have a calibration which can be applied, an informational message is displayed and calibration is turned off.

Suggestions: Use a full 2-Port calibration to be compatible with any S-Parameter.

EventID: E8020266 (hex)

Message: 615

" New calset created."

Severity: Informational message.

Further explanation: The newly created cal set will be automatically named and time stamped. If this is the beginning of a calibration procedure, the cal set will not be stored to memory until the calibration has completed successfully. The new cal set will be deleted if the calibration is canceled or does not otherwise complete successfully.

Suggestions: Informational

EventID: 68020267

Message: 617

The calset file: <x> appears to be corrupted and cannot be removed. Exit the application, remove the file, and restart.

Severity: Error

Suggestions: The cal set file is stored in the application home directory C:/Program Files/Agilent/Network Analyzer/PNACalSets.dat. Remove this file, then restart the application.

EventID: E8020269 (hex)

Message: 634

"The calset file: <x> load failed."

Severity: Error

Further explanation: The calset file contains a collection of calsets. The file resides on the hard drive.

Suggestions: Try restarting the application. If the failure persists, you may have to delete the cal set data file and restart the application. The cal set file is stored in the application home directory. C:/Program Files/Agilent/Network Analyzer/PNACalSets.dat. Remove this file, then restart the application.

EventID: E802027A (hex)

Message: 635

"The calset file: <x> save failed."

Severity: Error

Further explanation: The file operation detected an error. The save operation was aborted.

Suggestions: Retry.

EventID: E802027B (hex)

Message: 636

"A calset was deleted."

Severity: Informational

Further explanation: One of the calsets has been successfully deleted from the collection of calsets available. This can happen as the result of a user request or intentional operation.

Suggestions: None

EventID: 6802027C (hex)

Message: 637

"The version of the calset file: <x> is not compatible with the current instrument."

Severity: Error

Further explanation: A versioning error can prevent a calset from being used. This can happen as a result of instrument firmware upgrades.

Suggestions: If the versioning error is the result of firmware upgrade, you will have to re-install the old version of firmware to re-use the calset file. Or you can re-create the calsets with the current

version of firmware.

The cal set file is stored in the application home directory C:/Program Files/Agilent/Network Analyzer/PNACalSets.dat. Remove this file, then restart the application.

EventID: E802027D (hex)

Message: 638

"Incompatible CalSets found: <x> of <y> stored calsets have been loaded."

Severity: Error

Further explanation: Errors were found on some of the calsets stored in the calset file. The errors may have been caused by versioning issues that may have corrupted the various calset keys.

Suggestions: Use the calset viewer to look at the contents of calset files. Delete the files that are corrupted.

EventID: 6802027E (hex)

Message: 639

"The Calset file: <x> was not found. A new file has been created."

Severity: Informational

Further explanation: The calset file should be stored on the hard drive. When the application is started, a search is done and the file is loaded if it can be found. If the file is not found, the analyzer will create a new file and display this message.

Suggestions: None

EventID: 6802027F (hex)

Message: 640

"The Calset specified is currently in use."

Severity: Error

Further explanation: This may indicate a conflict between multiple calset users attempting calibration tasks.

Suggestions: Save the instrument state. Preset the analyzer and recall the instrument state. This may abort any processes that may be in progress.

EventID: E8020280 (hex)

Message: 641

"The calset specified has not been opened."

Severity: Error

Further explanation: Multiple users may be attempting to access the calset.

Suggestions: Close multiple calset users so that only one user will access the calset.

EventID: E8020281 (hex)

Message: 642

"The maximum number of cal sets has been reached. Delete old or unused cal sets before attempting to create new ones."

Severity: Error

Suggestions: You may also delete the calsets data file.

The cal set file is stored in the application home directory. C:/Program Files/Agilent/Network_Analyzer/PNACalSets.dat. Remove this file, then restart the application.

EventID: E8020282 (hex)

Message: 643

The requested power loss table segment was not found.

Severity: Error

EventID: E8020283 (hex)

Message: 644

"A valid calibration is required before correction can be turned on."

Severity: Error

Further explanation: This usually indicates a calibration procedure has not run to completion or that the selected measurement does not have a valid calibration available from within the currently selected cal set.

Suggestions: To find a calibration, you can select a Cal Set that contains the calibration needed for the current measurements. This error can happen if a calibration type is not specified before attempting to

programmatically execute cal acquisitions. Specify the cal type before beginning a calibration procedure.

EventID: E8020284 (hex)

Message: 645

The cal data for <x> is incompatible and was not restored. Please recalibrate."

Severity: Warning

Further explanation: None

Suggestions: None

EventID: A8020285 (hex)

Message: 646

"CalSet not loaded, version is too new."

Severity: Error

Further explanation: An old version of firmware is attempting to run with a new calset version. The version is incompatible.

Suggestions: The calset can be removed. You may also delete the calsets data file if you are migrating between various firmware revisions often and you would like to avoid this error. The cal set file is stored in the application home directory. C:/Program Files/Agilent/Network Analyzer/PNACalSets.dat. Remove this file, then restart the application.

EventID: E8020286 (hex)

Message: 647

"Custom cal type not found."

Severity: Error

Further explanation:

Suggestions:

EventID: E8020287 (hex)

Message: 648

"Custom correction algorithm defers to the client for interpolation."

Severity: Informational

EventID: 68020288 (hex)

Message: 649

"Custom cal dll threw an exception."

Severity: Error

EventID: E8020289 (hex)

Message: 650

"Could not load the ecal.dll library"

Severity: Error

EventID: E802028A (hex)

Message: 656

"The argument specified is not a valid cal type."

Severity: Error

EventID: E8020290 (hex)

Message: 657

"The function found existing interpolated data"

Severity: Informational

EventID: 68020291 (hex)

Message: 658

"The function computed new interpolation values."

Severity: Informational

EventID: 68020292 (hex)

Message: 659

"The source power measurement failed."

Severity: Error

Suggestions: Please check GPIB, power meter settings and sensor connections.

EventID: E8020293 (hex)

Message: 660

"Duplicate session found. Close session and retry."

Severity: Error

EventID: E8020294 (hex)

Message: 661

"The session does not exist. Open the session and try again."

Severity: Error

Further explanation:

EventID: E8020295 (hex)

Message: 662

"Attempt to launch a custom calibration failed."

Severity: Error

Further explanation:

EventID: E8020296 (hex)

Message: 663

"Request to measure a cal standard failed."

Severity: Error

Further explanation: Please ensure you are requesting to measure standards which are defined for this calibration.

EventID: E8020297 (hex)

Message: 664

"Since Electronic Calibration Kit is selected, Mechanical Cal Kit parameter cannot be changed."

Severity: Error

Further explanation:

EventID: E8020298 (hex)

Message: 665

"Frequencies of the active channel are below minimum or above maximum frequencies of the ECal module factory characterization."

Suggestions: Change the channel frequencies, or select another ECal module.

Severity: Error

EventID: E8020299 (hex)

Message: 666

"Calset chosen for characterizing the ECal Module Ports %1 does not contain a calibration for PNA Ports %2."

Severity: Error

Suggestions: Go back to select another calset or to perform another cal.

EventID: E802029A (hex)

Message: 667

"ECal module only has sufficient memory remaining to store a maximum of %1 points in User Characterization %2."

Severity: Error

Suggestions: Decrease your number of points, or choose to overwrite another user characterization.

EventID: E802029B (hex)

Message: 668

Input values are non-monotonic. Cannot interpolate.

Severity: Error

EventID: E802029C (hex)

Message: 669

Interpolation target is out of range. Cannot interpolate.

Severity: Error

EventID: E802029D (hex)

Message: 670

Guided Calibration Error: <>

Severity: Error

EventID: E802029E (hex)

Message: 671

The first call to the guided calibration interface must be Initialize.

Severity: Error

EventID: E802029F (hex)

Message: 672

The selected thru cal method was not recognized.

Severity: Error

EventID: E80202A0 (hex)

Message: 673

Could not generate the error terms.

Severity: Error

EventID: E80202A1 (hex)

Message: 674

Guided calibration must be performed on the active channel

Severity: Error

EventID: E80202A2 (hex)

Message: 675

You can not start using calibration steps until you have successfully called [generate steps](#).

Severity: Error

EventID: E80202A3 (hex)

Message: 676

The step number given is out of range. Step numbers should be between 1 and the number of steps. 0 is not a valid step number.

Severity: Error

EventID: E80202A4 (hex)

Message: 677

A calset was selected for channel: <n> without restoring stimulus.

Severity: Informational

EventID: 680202A5 (hex)

Message: 678

A calset was selected for channel: <n> restoring stimulus.

Severity: Informational

EventID: 680202A6 (hex)

Message: 679

The selected calset stimulus could not be applied to the channel.

Severity: Informational

EventID: 680202A7 (hex)

Message: 680

You attempted to measure power at a frequency outside the frequency range defined for the specified power sensor. Select another sensor or adjust the range for this sensor.

Severity: Error

EventID: E80202A8 (hex)

Message: 681

Specified frequency is outside the frequency ranges currently defined for the power meter's sensors.

Severity: Error

EventID: E80202A9 (hex)

Message: 682

Additional Calibration Standards need to be acquired in order to calibrate over the entire frequency range currently being measured.

Severity: Informational

EventID: 680202AA (hex)

Message: 683

The PNA failed to convert cal kits for use by unguided calibrations. The recommended action is to restore Cal Kit defaults.

Severity: Error

EventID: E80202AB (hex)

Message: 684

The PNA failed to convert cal kits for use by unguided calibrations. CalKit defaults have been restored.

Severity: Error

EventID: E80202AC (hex)

Message: 685

Power meter is reserved by a source power cal acquisition already in progress.

Severity: Error

EventID: E80202AD (hex)

Message: 686

Source power calibration has not been performed or uploaded for the specified channel and source port.

Severity: Error

EventID: E80202AE (hex)

Message: 687

Source power calibration data array size for the specified channel and source port does not match it's associated stimulus number of points.

Severity: Error

EventID: E80202AF (hex)

Message: 688

Source power calibration of Port <n> on Channel <n> was turned off because the correction array no longer exists.

Severity: Error

EventID: E80202B0 (hex)

Message: 689

This command can only be used on a measurement created with a specified calibration loadport.

Severity: Error

EventID: E80202B1 (hex)

Message: 690

Interpolation is turned off and you have changed the stimulus settings of the original calibration, so correction has been turned off.

Severity: Error

EventID: E80202B2 (hex)

Message: 691

Stimulus settings for your current instrument state exceeded the parameters of the original calibration, so correction has been turned off.

Severity: Error

EventID: E80202B3 (hex)

Message: 692

Fixturing: the requested S2P file cannot be read. Possible formatting problem.

Severity: Error

EventID: E80202B4 (hex)

Message: 693

Fixturing: the requested S2P file cannot be opened.

Severity: Error

EventID: E80202B5 (hex)

Message: 694

Fixturing: the requested S2P file cannot be interpolated. This is usually because the frequency range in the file is a subset of the current channel frequency range.

Severity: Error

EventID: E80202B6 (hex)

Message: 695

Cal Registers can only be used by one channel: the channel conveyed in the name of the cal register. The name cannot be changed.

Severity: Error

Further explanation: See [Cal Registers](#)

EventID: E80202B7 (hex)

Message: 696

Fixturing: cannot be enabled with Response Calibrations and has been turned off.

Severity: Error

EventID: E80202B8 (hex)

Message: 697

The selected calibration cannot be performed for this measurement.

Severity: Error

EventID: E80202B9 (hex)

Message: 698

Fitting: RemoveAllConnectors() should be called prior to calling AddConnector after a fit has been attempted.

Severity: Error

EventID: E80202BA (hex)

Message: 699

An attempt was made to acquire calibration data before the system was properly initialized.

Severity: Error

EventID: E80202BB (hex)

Message: 700

Use IGuidedCalibration for multiport calibration types.

Severity: Error

EventID: E80202BC (hex)

Message: 701

Guided calibration requires number of thru measurement paths be at least equal to the number of calibration ports minus 1.

Severity: Error

EventID: E80202BD (hex)

Message: 702

A thru path was specified that includes a port which the calibration was not specified to include.

Severity: Error

EventID: E80202BE (hex)

Message: 703

One or more of the ports to be calibrated was not found in the set of specified thru paths.

Severity: Error

EventID: E80202BF (hex)

Hardware Errors

Message: 770

Input power too high. Source power is off.

Severity: Warning

EventID: A8030302 (hex)

Message: 771

Source power restored.

Severity: Informational

EventID: 68030303 (hex)

Message: 772

"The spampnp.sys driver is not working. Check system hardware. ! Data will be simulated. !"

Severity: Error

Further explanation: The Network Analyzer application cannot locate the DSP board. Hardware or a driver may be malfunctioning. This is also common when attempting to run the Network Analyzer on a workstation.

EventID: E8030304 (hex)

Message: 773

"Instrument Serial Bus Not Working."

Severity: Error

Further explanation: The instrument EEPROM appears to contain either all ones or all zeros. A serial bus hardware failure prevents reading the EEPROM.

EventID: E8030305 (hex)

Message: 784

Unleveled, source <n>, out <n>.

Severity: Error

Further explanation: The PNA was unable to set the power on port <n> to the desired level

Message: 848

"Phase lock lost"

Severity: Error

Further explanation: The instrument source was not able to lock properly. This can be the result of broken hardware, poor calibration, or bad EEPROM values.

Suggestions: Perform source calibration. Click System / Service / Adjustments / Source Calibration

EventID: E8030350 (hex)

Message: 849

Phaselock restored.

Severity: Success

EventID: 0x28030351 (hex)

Message: 850

"Unknown hardware error."

Severity: Error

Further explanation: Hardware malfunctioned prevents communication with the DSP.

EventID: E8030352 (hex)

Message: 851

DSP communication lost.

Severity: Error

EventID: E8030353 (hex)

Message: 852

RF power off.

Severity: Error

EventID: E8030354 (hex)

Message: 853

RF power on.

Severity: Success

EventID: 28030355 (hex)

Message: 854

Hardware OK.

Severity: Success

EventID: 28030356 (hex)

Message: 855

"Source unlevelled."

Severity: Error

Further explanation: The source was unable to properly level at the requested power. The indicated power may not be accurate.

Suggestions: Try a different power level. Recalibrate source, if problem persists.

EventID: E8030357 (hex)

Message: 856

Source leveled.

Severity: Success

EventID: 28030358 (hex)

Message: 857

Input overloaded.

Severity: Error

EventID: E8030359 (hex)

Message: 858

Input no longer overloaded.

Severity: Success

EventID: 2803035A (hex)

Message: 859

"Yig calibration failed."

Severity: Error

Further explanation: Internal self-calibration of YIG oscillator tuning failed.

EventID: E803035B (hex)

Message: 860

Yig calibrated.

Severity: Success

EventID: 2803035C (hex)

Message: 861

"Analog ramp calibration failed."

Severity: Error

Further explanation: Internal analog sweep ramp calibration has failed.

EventID: E803035D (hex)

Message: 862

Analog ramp calibrated.

Severity: Success

EventID: 2803035E (hex)

Message: 863

Source temperature high.

Severity: Error

EventID: E803035F (hex)

Message: 864

Source temperature OK.

Severity: Success

EventID: 28030360 (hex)

Message: 865

"EEPROM write failed."

Severity: Error

Further explanation: Attempt to store calibration data to EEPROM has failed. There is a possible hardware failure.

EventID: E8030361 (hex)

Message: 866

EEPROM write succeeded.

Severity: Success

EventID: 28030362 (hex)

Message: 867

Attempted I/O write while port set to read only.

Severity: Error

Further explanation: Attempt to write to an I/O data port while the port set to input/read only.

Suggestions: Set data port to write/output before attempting to write to port.

EventID: E8030363 (hex)

Message: 868

" Attempted I/O read from write only port.

Severity: Error

Further explanation: Attempt to read from an I/O data port while the port set to output/write only.

Suggestions: Set data port to read/input before attempting to read from port.

EventID: E8030364 (hex)

Message: 869

Invalid hardware element identifier.

Severity: Error

EventID: E8030365 (hex)

Message: 870

Invalid gain level setting.

Severity: Error

EventID: E8030366 (hex)

Message: 871

Device driver was unable to allocate enough memory. Please try rebooting.

Severity: Error

EventID: E8030367 (hex)

Message: 872

DSP Error. Please Contact Agilent Support. Technical Information: DSP Type 1

Severity: Error

EventID: E8030368 (hex)

Message: 873

DSP Error. Please Contact Agilent Support. Technical Information: DSP Type 2

Severity: Error

EventID: E8030369 (hex)

Message: 874

DSP Error. Please Contact Agilent Support. Technical Information: DSP Type 3

Severity: Error

EventID: E803036A (hex)

Message: 875

DSP Error. Please Contact Agilent Support. Technical Information: DSP Type 4

Severity: Error

EventID: E803036B (hex)

Message: 876

DSP Error. Please Contact Agilent Support. Technical Information: DSP Type 5

Severity: Error

EventID: E803036C (hex)

Message: 910

The trigger connection argument was not recognized as valid by the firmware.

Severity: Error

EventID: 0xE803038E (hex)

Message: 911

The trigger connection specified does not support this trigger behavior

Severity: Error

EventID: E803038F (hex)

Message: 912

The trigger behavior specified was not recognized as valid by the firmware.

Severity: Error

EventID: E8030390 (hex)

Message: 913

The trigger connection specified does not physically exist on this network analyzer

Severity: Error

EventID: E8030391 (hex)

Message: 914

Cannot set "Accept Trigger Before Armed", since this hardware configuration does not support edge triggering.

Severity: Error

EventID: E8030392 (hex)

Message: 915

Cannot set "Trigger Output Enabled", since this hardware configuration does not support BNC2.

Severity: Error

EventID: E8030393 (hex)

Message: 916

Exceeded maximum trigger delay.

Severity: Error

EventID: E8030394 (hex)

Message: 917

Exceeded minimum trigger delay.

Severity: Error

EventID: E8030395 (hex)

Measure Errors

Message: 1024

If you are going to display or otherwise use a memory trace, you must first store a data trace to memory.

Severity: Warning

EventID: A8040400 (hex)

Message: 1025

"The measurement failed to shut down properly. The application is in a corrupt state and should be shut down and restarted."

Severity: Error

Further explanation: This message is displayed if the PNA application becomes corrupt. If you continue to get this error, please call customer service

EventID: E8040401 (hex)

Message: 1026

The measurement failed to shut down properly. The update thread failed to exit properly.

Severity: Warning

EventID: A8040402 (hex)

Message: 1027

"Group Delay format with CW Time or Power sweeps produces invalid data."

Severity: Warning

Further explanation: Group Delay format is incompatible with single-frequency sweeps. Invalid data is produced.

Suggestions: Ignore the data or choose a different format or sweep type.

EventID: A8040403 (hex)

Message: 1028

Severity: Informational

"MSG_LIMIT_FAILED"

Further explanation: Limit line test failed.

EventID: 68040404 (hex)

Message: 1029

Severity: Informational

"MSG_LIMIT_PASSED"

Further explanation: Limit line test passed.

EventID: 68040405 (hex)

Message: 1030

"Exceeded the maximum number of measurements allowed."

Severity: Warning

Further explanation: See [Traces, Channels, and Windows on the PNA](#) for learn about maximum measurements.

EventID: A8040406 (hex)

Message: 1031

"Network Analyzer Internal Error. Unexpected error in AddNewMeasurement."

Severity: Warning

Further explanation: If you continue to get this message, contact product support.

EventID: A8040407 (hex)

Message: 1032

"No measurement was found to perform the selected operation. Operation not completed."

Severity: Warning

Further explanation: None

Suggestions: Create a measurement before performing this operation.

EventID: A8040408 (hex)

Message: 1033

The Markers All Off command failed.

Severity: Warning

EventID: A8040409 (hex)

Message: 1034

"A memory trace has not been saved for the selected trace. Save a memory trace before attempting

trace math operations."

Severity: Warning

Further explanation: Must have a memory trace when trying to do Trace Math,

EventID: A804040A (hex)

Message: 1035

"MSG_SET_AVERAGE_COMPLETE"

Severity: Informational

Further explanation: Informational for COM programming. Averaging factor has been reached.

EventID: 6804040B (hex)

Message: 1036

"MSG_CLEAR_AVERAGE_COMPLETE"

Further explanation: Informational for COM programming. Averaging factor has NOT been reached.

EventID: 6804040C (hex)

Message: 1037

"Time Domain transform requires at least 3 input points. The transform has been deactivated."

Severity: Informational

Further explanation: None

Suggestions: Increase the number of points.

EventID: 6804040D (hex)

Message: 1038

Smoothing requires a scalar format, and has been deactivated.

Severity: Informational

EventID: 6804040E (hex)

Message: 1039

A receiver power calibration in this instrument state file cannot be recalled into this firmware version.

Severity: Warning

EventID: A804040F (hex)

Message: 1047

Could not achieve target power.

Severity: Error

Further explanation: This indicates that the PNA was unable to find a source power during the THRU step of the cal sufficiently high to boost the measured noise power on port 2 to 6 dB above the noise floor.

Message: 1104

"Exceeded limit on number of measurements."

Severity: Error

Further explanation: See [Traces, Channels, and Windows on the PNA](#) for measurement limits.

EventID: E8040450 (hex)

Message: 1105

"Parameter not valid."

Severity: Error

Further explanation: A measurement parameter that was entered programmatically is not valid.

EventID: E8040451 (hex)

Message: 1106

"Measurement not found."

Severity: Error

Further explanation: Any of these could be the cause:

Trying to calibrate but already have maximum measurements.

Trying to do a confidence check but there is not a measurement.

Trying to create, activate, or alter a measurement through COM that has been deleted through the front

panel.

Trying to use a trace name through programming that is not unique.

EventID: E8040452 (hex)

Message: 1107

"No valid memory trace."

Severity: Error

Further explanation: Must have a memory trace when trying to do Trace Math,

Suggestions: Store a memory trace.

EventID: E8040453 (hex)

Message: 1108

"The reference marker was not found."

Severity: Error

Further explanation: Attempted to create a delta marker without first creating a reference marker (COM only).

EventID: E8040454 (hex)

Message: 1109

"Data and Memory traces are no longer compatible. Trace Math has been turned off."

Severity: Error

Further explanation: Warning - channel setting has changed while doing trace math.

Suggestions: Store another memory trace and turn trace math back on.

EventID: A8040455 (hex)

Message: 1110

"Data and Memory traces are not compatible. For valid trace math operations, memory and data traces must have similar measurement conditions."

Severity: Error

Further explanation: Tried to do trace math without compatible data and memory traces.

Suggestions: Store another memory trace.

EventID: E8040456 (hex)

Message: 1111

"Marker Bandwidth not found."

Severity: Error

Further explanation: Could not find a portion of trace that meets the specified bandwidth criteria.

EventID: E8040457 (hex)

Message: 1112

"The peak was not found."

Severity: Error

Further explanation: Could not find portion of trace that meets peak criteria.

Suggestions: See Marker Peak criteria.

EventID: E8040458 (hex)

Message: 1113

"The target search value was not found."

Severity: Error

Further explanation: Could not find interpolated data point that meets search value.

EventID: E8040459 (hex)

Message: 1114

"Reflection measurement, such as S11, must supply an auxiliary port to disambiguate 2-port measurements on multipoint instruments."

Severity: Error

Further explanation:

EventID: E804045A (hex)

Message: 1115

"The receiver power calibration has been turned off because the type of measurement or source port has changed, so the calibration is no longer valid."

Severity: Warning

Further explanation:

EventID: A804045B (hex)

Message: 1116

"Receiver power cal requires the active measurement to be of unratiod power."

Severity: Warning

Further explanation:

EventID: A804045C (hex)

Message: 1117

"There is currently no source power calibration associated with the channel and source port of the active measurement. A source power cal should be performed or recalled before performing a receiver power calibration."

Severity: Warning

Further explanation:

EventID: A804045D (hex)

Message: 1118

"The attempted operation can only be performed on a standard measurement type."

Severity: Error

Further explanation:

EventID: E804045E (hex)

Message: 1119

"The custom measurement cannot be loaded because it is not compatible with the Network Analyzer hardware."

Severity: Error

Further explanation:

Suggestions:

EventID: E804045F (hex)

Message: 1120

"The custom measurement cannot be loaded because it is not compatible with the Network Analyzer software."

Severity: Error

Further explanation:

EventID: E8040460 (hex)

Message: 1121

"The custom measurement load operation failed for an unspecified reason."

Severity: Error

Further explanation:

EventID: E8040461 (hex)

Message: 1122

"The custom measurement data processing has generated an unhandled exception, and will be terminated. The PNA software may be in an unstable state and it is recommended that the PNA software be shutdown and restarted."

Severity: Error

Further explanation:

EventID: E8040462 (hex)

Message: 1123

"The attempted operation can only be performed on a custom measurement type."

Severity: Error

Further explanation:

EventID: E8040463 (hex)

Message: 1124

"The requested custom measurement is not available."

Severity: Error

Further explanation:

EventID: E8040464 (hex)

Message: 1125

"The requested custom algorithm was not found."

Severity: Error

Further explanation:

EventID: E8040465 (hex)

Message: 1126

"Normalization cannot be turned on because the measurement does not have a valid divisor buffer."

Severity: Error

Further explanation:

EventID: E8040466 (hex)

Message: 1127

"The Raw Data requested by the measurement could not be provided."

Severity: Warning

Further explanation:

EventID: A8040467 (hex)

Message: 1128

"The selected Sweep Type does not allow Transform and Gating. Transform and Gating disabled. "

Severity: Error

Further explanation:

EventID: E8040468 (hex)

Message: 1129

Memory trace can not be applied to this measurement

Severity: Error

EventID: E8040469 (hex)

Message: 1130

Normalization can not be applied to this measurement

Severity: Error

EventID: E804046A (hex)

Message: 1131

The data provided has an invalid number of points. It could not be stored

Severity: Error

EventID: E804046B (hex)

Message: 1132

The measurement stored in the save/recall state has an invalid version. It could not be loaded

Severity: Error

EventID: E804046C (hex)

Message: 1133

This data format argument for this operation must be "naDataFormat_Polar"

Severity: Error

EventID: E804046D (hex)

Message: 1134

This data format argument for this operation must be a scalar data format

Severity: Error

EventID: E804046E (hex)

Message: 1135

The memory trace is not valid for the current measurement setup.

Severity: Error

EventID: E804046F (hex)

Message: 1136

This measurement is incompatible with existing measurements in this channel. Choose another channel.

Severity: Error

EventID: E8040470 (hex)

Message: 1137

Port extension correction is not available for offset frequency measurements. Port extension correction has been disabled.

Severity: Error

EventID: E8040471 (hex)

Message: 1138

Physical port number assignments for logical port mappings must be unique.

Severity: Error

EventID: E8040472 (hex)

Parser Errors

Message: 1281

"You have sent a read command to the analyzer without first requesting data with an appropriate output command. The analyzer has no data in the output queue to satisfy the request."

Severity: Error

EventID: 68050501 (hex)

Message: 1282

"You must remove the active controller from the bus or the controller must relinquish the bus before the analyzer can assume the system controller mode."

Severity: Error

EventID: E8050502(hex)

Message: 1283

"The analyzer did not receive a complete data transmission. This is usually caused by an interruption of the bus transaction."

Severity: Error

EventID: E8050503 (hex)

Message: 1284

"The instrument status byte has changed."

Severity: Informational

EventID: 68050504 (hex)

Message: 1285

"The SCPI command received has caused error number %1: "%2"."

Severity: Informational

EventID: 68050505 (hex)

Message: 1286

"The INET LAN server has been started as process number %1."

Severity: Informational

EventID: 68050506 (hex)

Message: 1360

"Execution of the SCPI command has failed"

Severity: Error

EventID: E8050550 (hex)

Message: 1361

" The INET/LAN device is not accessible."

Severity: Error

EventID: E8050551 (hex)

Message: 1362

"The INET/LAN driver was not found. "

Severity: Error

EventID: E8050552 (hex)

Message: 1363

"The INET/LAN driver was not found."

Severity: Error

EventID: E8050553 (hex)

Message: 1364

"The INET/LAN device is unable to acquire the necessary resources. "

Severity: Error

EventID: E8050554 (hex)

Message: 1365

"The INET/LAN device generated a generic system error. "

Severity: Error

EventID: E8050555 (hex)

Message: 1366

"Invalid address for the INET/LAN device."

Severity: Error

EventID: E8050556 (hex)

Message: 1367

"The INET I/O library was not found. "

Severity: Error

EventID: E8050557 (hex)

Message: 1368

"An error occurred in the INET system. "

Severity: Error

EventID: E8050558 (hex)

Message: 1369

"Access to the INET/LAN driver was denied. "

Severity: Error

EventID: E8050559 (hex)

Message: 1370

"Could not load error system message dll."

Severity: Error

EventID: E805055A (hex)

Message: 1371

"ErrorSystemMessage.dll does not export the right function."

Severity: Error

EventID: E805055B (hex)

Message: 1372

"Custom scpi library was not able to be knitted"

Severity: Error

EventID: E805055C (hex)

Message: 1373

"Could not knit the scpi error messages from the ErrorSystemMessage lib"

Severity: Error

EventID: E805055D (hex)

Message: 1374

Command is obsolete with this software version.

Severity: Error

EventID: E808055E (hex)

Message: 1375

CALC measurement selection set to none. Use [Calc:Par:Sel](#)

Severity: Error

EventID: E808055F (hex)

Message: 1535

"Parser got command: %1."

Severity: Informational

EventID: 680505FF (hex)

Display Errors 1536 - 1621

Message: 1536

"Exceeded the maximum of 4 traces in each window. The trace for <x> will not be added to window <x>."

Severity: Warning

Further explanation: None

Suggestions: Create the trace in another window. See the [PNA window limits](#).

EventID: A8060600 (hex)

Message: 1537

"Exceeded the maximum of 16 data windows. New window will not be created."

Severity: Warning

Further explanation: None

Suggestions: Create the trace in an existing window. See the [PNA window limits](#).

EventID: A8060601 (hex)

Message: 1538

"No Data Windows are present. Unable to complete operation."

Severity: Warning

Further explanation: Your remote SCPI operation tried to create a new measurement while there were no windows present

Suggestions: Create a new window before creating the measurement. See example [Create a measurement using SCPI](#)

EventID: A8060602 (hex)

Message: 1539

"No data traces are present in the selected window. Operation not completed."

Severity: Warning

Further explanation: None

EventID: A8060603 (hex)

Message: 1540

"Cannot complete request to arrange existing measurements in <x> windows due to the limit of <x> traces per window."

Severity: Informational

Further explanation: The arrange window feature cannot put the existing traces into the number of windows you requested because only 4 traces per window are allowed. [See Arranging Existing Measurements](#)

Suggestions: Either create more windows or delete some traces.

EventID: 68060604 (hex)

Message: 1541

"Unable to establish a connection with the specified printer."

Severity: Warning

Further explanation: None

Suggestions: Refer to Printer Help

EventID: A8060605 (hex)

Message: 1542

"Printout canceled."

Severity: Informational

EventID: 68060606 (hex)

Message: 1616

"Window not found."

Severity: Error

Further explanation: A window was specified in your program which does not exist.

Suggestions: Query the name of your window before specifying.

EventID: E8060650 (hex)

Message: 1617

"Duplicate window ID specified."

Severity: Error

Further explanation: None

EventID: E8060651 (hex)

Message: 1618

"Exceeded limit on number of windows."

Severity: Error

Further explanation: There is a limit of 4 windows per screen.

EventID: E8060652 (hex)

Message: 1619

"Exceeded limit on number of traces/window."

Severity: Error

Further explanation: There is a limit of 4 traces per window. See the [Traces, Channels, and Windows on the PNA](#).

Suggestions: Create the trace in another window

EventID: E8060653 (hex)

Message: 1620

"Trace not found."

Severity: Error

Further explanation: Your program tried to communicate with a non-existing trace.

Suggestions: Query the trace ID before writing to it.

EventID: E8060654 (hex)

Message: 1621

"The operating system does not recognize this printer."

Severity: Warning

EventID: A8060655 (hex)

Message: 1622

Duplicate trace ID specified.

Severity: Error

EventID: E8060656 (hex)

Channel Errors 1792 -1878

Message: 1792

"Sweep Complete."

Severity: Informational

Further explanation: None

Suggestions:None

EventID: 68070700 (hex)

Message: 1793

"All triggerable acquisitions have completed."

Severity: Informational

Further explanation:

EventID: 68070701 (hex)

Message: 1794

"The last trigger produced an aborted sweep."

Severity: Informational

Further explanation:

EventID: 68070702 (hex)

Message: 1795

"The segment list must be adjusted to have at least one active segment with more than 0 points to use segment sweep."

Severity: Informational

Further explanation: You attempted to change **Sweep type** to Segment sweep, but there is either no segments defined or no sweep points in the defined segments

Suggestions: Define at least one segment with at least one measurement point. See Segment sweep for more information

EventID: 68070703 (hex)

Message: 1796

"MSG_SET_CHANNEL_DIRTY"

Severity: Informational

Further explanation: This informational message occurs when a channel setting has changed but the channel still has data that was taken with the previous setting. The following CLEAR message occurs when new channel data is taken.

EventID: 68070704 (hex)

Message: 1797

"MSG_CLEAR_CHANNEL_DIRTY"

Severity: Informational

Further explanation: The previous SET message occurs when a channel setting has changed but the channel still has data that was taken with the previous setting. This CLEAR message occurs when new channel data is taken.

EventID: 68070705 (hex)

Message: 1798

Sweep time has changed from Auto to Manual mode. If desired to return to Auto mode, enter sweep time value of 0.

Severity: Informational

EventID: 68070706 (hex)

Message: 1799

"Set Sweep Completed"

Severity: Informational

Further explanation: This event occurs when a sweep and it's associated sweep calculations finish. This is typically when all sweeps on a channel complete.

EventID: 68070707 (hex)

Message: 1800

"Clear Sweep Completed"

Severity: Informational

Further explanation: This event occurs immediately after the SET SWEEP COMPLETED event. These two events set and clear the "Sweep Completed" bit (bit 4) on the SCPI Device Status register.

EventID: 68070708 (hex)

Message: 1801

"All Sweeps Completed and Processed"

Severity: Informational

Further explanation: This event occurs when all of the sweeps and sweep calculations are complete for a channel.

EventID: 68070709 (hex)

Message: 1802

Low Pass : Frequency limits have been changed.

Severity: Informational

EventID: 6807070A (hex)

Message: 1803

Low Pass : Number of points have been changed.

Severity: Informational

EventID: 6807070B (hex)

Message: 1804

Low Pass : Frequency limits and number of points have been changed.

Severity: Informational

EventID: 6807070C (hex)

Message: 1805

"Channel created"

Severity: Informational

EventID: 6807070D (hex)

Message: 1806

"Channel deleted"

Severity: Informational

EventID: 6807070E (hex)

Message: 1872

"Channel not found."

Severity: Error

Further explanation: A non-existent channel is being referenced under program control.

Suggestions: Query the channel number, then refer to it by number.

EventID: E8070750 (hex)

Message: 1873

"The requested sweep segment was not found."

Severity: Error

Further explanation: A non-existent sweep segment is being referenced under program control.

EventID: E8070751 (hex)

Message: 1874

"The sweep segment list is empty."

Severity: Error

Further explanation: Segment Sweep cannot be specified unless there is at least one defined segment. This error will only occur under remote control.

EventID: E8070752 (hex)

Message: 1875

"The number of points in active sweep segment list segments is 0."

Severity: Error

Further explanation: Segment Sweep cannot be specified unless there is at least data point specified in a segment. This error will only occur under remote control.

EventID: E8070753 (hex)

Message: 1876

"The specified source attenuator is not valid."

Severity: Error

Further explanation: You tried to set the Attenuator property on the Channel object on a PNA that doesn't have a source attenuator.

EventID: E8070754 (hex)

Message: 1877

"Log Frequency sweep cannot be selected with the current Number of Points. Please reduce Number of Points."

Severity: Error

Further explanation: The maximum number of points that can be used for Log sweep is 401.

EventID: E8070755 (hex)

Message: 1878

"The requested Number of Points is greater than can be selected for Log Frequency sweep."

Severity: Error

Further explanation: The maximum number of points that can be used for Log sweep is 401.

EventID: E8070756 (hex)

Message: 1879

"Response frequencies exceeded instrument range so Frequency Offset has been turned off."

Severity: Error

Further explanation: This error is returned whenever the instrument detects that the stimulus sweep setup and Frequency Offset settings result in computed response frequencies that exceed instrument limits. When this occurs, the instrument automatically turns off Frequency Offset to avoid the out-of-range conditions.

Suggestions: When this condition has occurred, change settings for either the stimulus frequencies or Frequency Offset so that the Response frequencies are within instrument bounds. Once this is done, Frequency Offset can once again be turned on.

EventID: E8070757 (hex)

Message: 1880

The total number of points for all the given segments exceeds the maximum number of points supported. The segments were not changed.

Severity: Error

EventID: E8070758 (hex)

Message: 1881

This instance of the Channels object was not used to place the channels in Hold, so no channels were resumed.

Severity: Error

EventID: E8070759 (hex)

Message: 1882

The port number was outside the range of allowed port numbers.

Severity: Error

EventID: E807075A (hex)

Message: 1883

More ports than are present are required for this operation.

Severity: Error

EventID: E807075B (hex)

General Errors

Message: 2048

"The function you requested requires a capability provided by an option to the standard analyzer. That option is not currently installed."

Severity: Error

Further explanation: None

Suggestions: To view the options on your analyzer, click **Help / About Network Analyzer**. For more information see [PNA Options](#)

EventID: 68080800 (hex)

Message: 2049

"The feature you requested is not available on the current instrument."

Severity: Error

Further explanation: None

EventID: 68080801 (hex)

Message: 2050

"The feature you requested is incompatible with the current instrument state."

Severity: Error

Further explanation: None

Suggestions: None

EventID: 68080802 (hex)

Message: 2051

"File<x> has been saved."

Severity: Informational

Further explanation: None

EventID: 68080803 (hex)

Message: 2052

"Attempt to save <x> failed."

Severity: Error

Further explanation: None

Suggestions: If using a floppy disk, ensure it is inside the drive and the disk is not full. Check the filename for special characters.

EventID: E8080804 (hex)

Message: 2053

"Attempt to recall file failed because <x> was not found."

Severity: Error

Further explanation: None

EventID: E8080805 (hex)

Message: 2054

"<x> has a bad header."

Severity: Error

Further explanation: None

Suggestions: Recopy the file and / or delete the file.

EventID: E8080806 (hex)

Message: 2056

"Request to enter hibernate state."

Further explanation: None

EventID: 68080808 (hex)

Message: 2057

"Power up from automatic hibernate state. Program received PBT_APMRESUMEAUTOMATIC Message."

Further explanation: None

EventID: 68080809 (hex)

Message: 2058

"Power up from suspend hibernate state. Program received PBT_APMRESUMESUSPEND Message."

Further explanation: None

EventID: 6808080A (hex)

Message: 2059

"Power up from suspend hibernate state. Program received PBT_APMRESUMECRITICAL Message."

Severity: Warning

Further explanation: None

EventID: A808080B (hex)

Message: 2060

"Power up from unknown hibernate state UI recovery called. Program received no PBT_Message within the time allotted and is attempting recovery."

Severity: Warning

Further explanation: None

EventID: A808080C (hex)

Message: 2061

"<x> already exists. File is being overwritten."

Further explanation: Used only for remote applications

EventID: 6808080D (hex)

Message: 2062

"File has not been saved."

Severity: Error

Further explanation: Used only for remote applications

EventID: E808080E (hex)

Message: 2063

"File <x> has been recalled."

Further explanation: Used only for remote applications

EventID: 6808080F (hex)

Message: 2064

"State version in <x> is considered obsolete by this version of this code."

Severity: Error

Further explanation: You attempted to recall a file that is no longer valid.

Suggestions: You must recreate the file manually.

EventID: E8080810 (hex)

Message: 2065

"State version in <x> is newer than the latest version supported by this code."

Severity: Error

Further explanation: You attempted to recall a file that was created by a later version of the PNA application.

Suggestions: You must recreate the file manually.

EventID: E8080811 (hex)

Message: 2066

"Error occurred while reading file <x>"

Severity: Error

Further explanation: The file may be corrupt.

Suggestions: Try to recreate the file.

EventID: E8080812 (hex)

Message: 2067

"Windows shell error: <x>"

Severity: Error

Further explanation: None

EventID: E8080813 (hex)

Message: 2068

Send message timed out returning: <x>.

Severity: Error

Further explanation: None

EventID: E8080814 (hex)

Message: 2069

"Changing GPIB mode to System Controller."

Severity: Informational

Further explanation: None

EventID: 68080815 (hex)

Message: 2070

"Changing GPIB mode to Talker Listener."

Severity: Informational

Further explanation: None

EventID: 68080816 (hex)

Message: 2071

"The Network Analyzer can not be put in GPIB System Controller mode until the GPIB status is Local. Stop any remote GPIB programs which may be using the Network analyzer, press the Macro/Local key and try again. "

Severity: Informational

Further explanation: See [LCL and RMT Operation](#)

Suggestions: Press the Macro/Local key and try again.

EventID: 68080817 (hex)

Message: 2120

"This method can not be invoked through a late-bound COM call."

Severity: Error

Further explanation: None

Suggestions: Use the alternate method described in the [COM programming documentation](#)

EventID: E8080878 (hex)

Message: 2128

"The specified format is invalid."

Severity:Error

Further explanation: None

EventID: E8080850 (hex)

Message: 2129

"WINNT exception caught by Automation layer."

Severity: Error

Further explanation: None

EventID: E8080851 (hex)

Message: 2130

"Bad port specification."

Severity: Error

Further explanation: None

EventID: E8080852 (hex)

Message: 2131

"Failed to find a printer."

Severity: Error

Further explanation: None

Suggestions: See [Connecting to a Printer](#)

EventID: E8080853 (hex)

Message: 2132

"Manual trigger ignored."

Severity: Error

Further explanation: None

EventID: E8080854 (hex)

Message: 2133

"Attempt to set trigger failed."

Severity: Error

Further explanation: None

EventID: E8080855 (hex)

Message: 2134

"Macro execution failed."

Severity: Error

Further explanation: None

EventID: E8080856 (hex)

Message: 2135

"Specified macro definition is incomplete."

Severity: Error

Further explanation:

EventID: E8080857 (hex)

Message: 2137

"Block data length error."

Severity: Error

Further explanation: See [Getting Data from the Analyzer](#)

EventID: E8080859 (hex)

Message: 2139

"Requested data not found."

Severity: Error

Further explanation: None

EventID: E808085B (hex)

Message: 2142

"The parameter supplied was out of range, so was limited to a value in range before being applied to the instrument."

Severity: Success

Further explanation: None

Suggestions: View range limits before sending programming commands.

EventID: 2808085E (hex)

Message: 2143

The parameter supplied was out of range, so was limited to a value in range before being applied to the instrument.

Severity: Error

EventID: E808085F (hex)

Message: 2144

"Request failed. The required license was not found."

Severity: Error

Further explanation: None

EventID: E8080860 (hex)

Message: 2145

"A remote call to the front panel has returned hresult <x>"

Severity: Error

Further explanation: This may indicate a problem with the front panel

Suggestions: Contact Technical support

EventID: E8080861 (hex)

Message: 2146

The recall operation failed.

Severity: Error

Further explanation:

EventID: E8080862 (hex)

Message: 2147

Attempt to save file failed.

Severity: Error

Further explanation:

EventID: E8080863 (hex)

Message: 2148

Recall attempt failed because file was not found.

Severity: Error

Further explanation:

EventID: E8080864 (hex)

Message: 2149

Recall file has a bad header.

Severity: Error

Further explanation:

EventID: E8080865 (hex)

Message: 2150

Recall file version is obsolete and no longer compatible with this instrument.

Severity: Error

Further explanation:

EventID: E8080866 (hex)

Message: 2151

The recall file contains an istate version newer than this instrument. A remote call to the front panel has returned hresult %1

Severity: Error

Further explanation:

EventID: E8080867 (hex)

Message 2152

"Front Panel <x>

Severity: Error

Further explanation: None

EventID: E8080868 (hex)

Message 2153

"Front Panel message"

Severity: Informational

Further explanation: None

EventID: 68080869 (hex)

Message 2154

"Power Service <x>

Severity: Error

Further explanation: There is more than 1 instance of powerservice running. There should only be one running. This might happen after running install shield - especially when upgrading the CPU board.

Suggestions: Try rebooting. If this persists, please call [Customer Support](#).

EventID: E808086A (hex)

Message 2155

"Power Service <x>

Severity: Informational

Further explanation: None

EventID: 6808086B (hex)

Message 2156

"The Agilent Technologies GPIB driver can not be loaded or unloaded."

Severity: Error

Further explanation: None

Suggestions: If the problem persists, from the PNA desktop, right-click on My Computer. Click Properties, Click Hardware Tab. Click Device Manager Button. Expand GPIB Devices. Right-click and click Uninstall all GPIB interfaces devices. Reboot the PNA.

EventID: E808086C (hex)

Message 2157

"The National Instruments GPIB driver can not be loaded or unloaded."

Severity: Error

Further explanation: None

Suggestions: If the problem persists, from the PNA desktop, right-click on My Computer. Click Properties, Click Hardware Tab. Click Device Manager Button. Expand GPIB Devices. Right-click and click Uninstall all GPIB interfaces devices. Reboot the PNA.

EventID: E808086D (hex)

Message 2158

"The Agilent GPIB driver is loaded but it can not start its parser."

Severity: Error

Further explanation: None

EventID: E808086E (hex)

Message: 2159

The front panel is in remote mode.

Severity: Warning

EventID: A808086F (hex)

Message: 2160

The Registry Key specified could not be found.

Severity: Error

EventID: E8080870 (hex)

Message: 2161

An overcurrent condition has been detected on a probe plugged into the front panel.

Severity: Warning

EventID: A8080871 (hex)

Message: 2162

The operation timed out.

Severity: Error

EventID: E8080872 (hex)

Message 2163

"The Network Analyzer executed a preset."

Severity:Informational

Further explanation: None

EventID: 68080873 (hex)

Message 2164

"Access to file denied."

Severity: Error

Further explanation: This means that the system can not open an output file for writing. Most likely because the file is write protected.

Suggestions: Pick another file name or file directory, check floppy disk hard disk write access.

EventID: E8080874 (hex)

Message 2165

"File type is structured storage."

Severity: Informational

Further explanation: None

EventID: 68080875 (hex)

Message 2166

"The trigger operation failed."

Severity: Error

Further explanation: None

EventID: E8080876 (hex)

Message 2167

"Argument out of range error."

Severity: Error

Further explanation: None

Suggestions: None

EventID: E8080877 (hex)

Message: 2169

The given COM object is not a custom application

Severity: Error

EventID: E8080879 (hex)

Message: 2170

The eventID supplied was not recognized as a valid PNA eventID

Severity: Error

EventID: E808087A (hex)

Message: 2171

The operation was canceled.

Severity: Error

EventID: E808087B (hex)

Message: 2172

High security level cannot be disabled directly. Only an instrument preset or recall of lower security instrument state will reset this security level.

Severity: Error

EventID: E808087C (hex)

Message: 2173

Local lockout mode is on. The PNA application will not accept input from front panel, keyboard or mouse until this mode is turned off from a remote interface.

Severity: Error

EventID: E808087D (hex)

Message: 2174

The SnP request is not valid for the selected measurement.

Severity: Error

EventID: E808087E (hex)

Message: 2175

Preset is not supported while this dialog or wizard is open. Close the dialog or wizard and then try again.

Severity: Error

EventID: E808087F (hex)

Message: 2176

The function you requested requires a capability provided by an option to the standard analyzer. That option is not currently installed.

Severity: Error

EventID: E8080880 (hex)

Message: 2177

Catastrophic error. Crash dump recorded at <n>

Severity: Error

EventID: E8080881 (hex)

Message: 2178

In the context of a noise calibration, this would occur if the PNA was unable to set the state of the tuner Ecal module.

Severity: Error

EventID: E8080882 (hex)

Message: 2179

Failed to open gen.lic.

Severity: Error

EventID: E8080883 (hex)

Last modified:

Nov. 29, Updated bookmarks
2006

About Error Messages

PNA errors and Operating System errors are displayed and logged in an error file. You can choose how to display PNA errors, or choose to not display PNA errors at all.

- [Error Display](#)
- [View Error Log](#)
- [List of PNA Errors](#)
- [SCPI Errors](#)

Other System topics

Error Display

By default, error messages appear on the PNA screen for a brief period. You can choose to have them stay on the screen until you click an OK button, or have them not appear at all. When they stay on the screen, a Help button is available to provide further assistance.

How to select the display of Error Messages

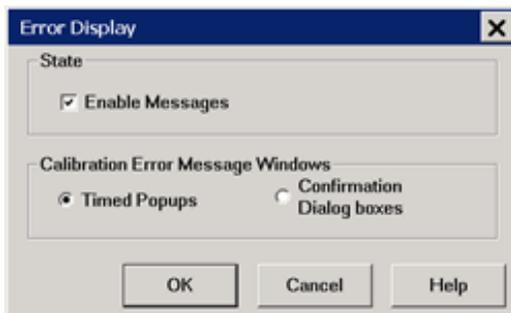
Using front-panel HARDKEY [softkey] buttons

1. Press **SYSTEM**
2. then [**Help**]
3. then [**Error Messages**]
4. then [**Error Display**]

Using a mouse with PNA Menus

1. Click **Help**
2. then **Error Messages**
3. then **Error Display**

◀ Programming Commands ▶



Error Display dialog box help

On Preset, these settings revert to their defaults (enabled, timed popups).

Enable Messages Check to display all PNA error messages as they occur. Clear to suppress the display of PNA error messages. You can still view them in the [error log](#).

Calibration Error Message Windows

Timed Popups Displays error messages on the screen for a duration of time proportional to the length of the message. You can then view the message in the [error log](#) and get further assistance.

Confirmation Dialog boxes Displays error messages in a standard dialog box. You then choose **OK** or **Cancel** to close the dialog box, or press **Help** to get further information on the error message.

View Error Log

The PNA Error Log is a list of all events that have occurred. (Events are used in programming the PNA using COM.) PNA errors is a subset of PNA events. Only events with severity codes of ERROR are displayed on the PNA screen as they occur. From the error log, you can access further help with an error by selecting the error and clicking Help.

How to view the Error Log

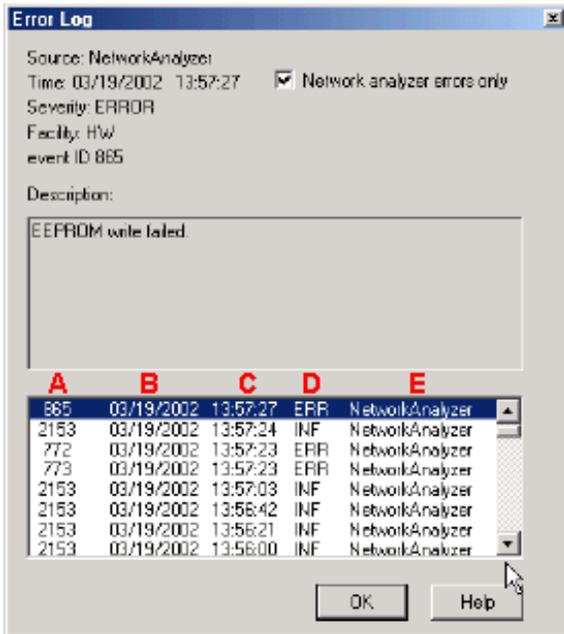
Using front-panel HARDKEY [softkey] buttons

1. Press **SYSTEM**
2. then **[Help]**
3. then **[Error Messages]**
4. then **[View Error Log]**

Using a mouse with PNA Menus

1. Click **Help**
2. then **Error Messages**
3. then **View Error Log**

No programming commands



Error Log dialog box help

Network analyzer errors only Select to view only PNA errors. Clear to view all errors that occur on all applications of the computer.

Description Error message that appears on the PNA screen.

A - Event ID Error message number

B - Date the Error occurred

C - Time the Error occurred

D - Severity Code - All events have one of the following severity codes:

- SUCcess - the operation completed successfully
- INFormational - events that occur without impact on the measurement integrity
- WARning - events that occur with potential impact on measurement integrity
- ERRor - events that occur with serious impact on measurement integrity

E - Application in which the error occurred.

OK Closes the Dialog box

Help Provides further information on the selected Error message

To clear the Error Log:

1. From the **File** menu click **Minimize Application**
2. On the desktop, select **Start, Settings, Control Panel**
3. On the Control Panel, click **Administrative Tools**
4. On the Administrative Tools window, click **Event Viewer**
5. On the Event Viewer window, right-click **Application**
6. Select **Clear all Events**
7. If you want to save a file with the contents of the Event Log, click **Yes**. Otherwise, click **No**

To restore the PNA application, click on the PNA Analyzer taskbar button at the bottom of the screen

Last Modified:

16-Sep-2010	Updated to PNA-X menu
24-Mar-2010	Updated to Error Display (A.09.20)
1-Jan-2007	MX Added UI

Analyzer Accessories

- [Coax Mechanical Calibration Kits](#)
- [Waveguide Mechanical Calibration Kits](#)
- [Electronic Calibration \(ECal\)](#)
- [Mechanical Verification Kits](#)
- [Adapter and Accessory Kits](#)
- [Test Port Cables](#)
- [USB Peripherals](#)
- [Connector Care and ESD Supplies](#)

Other Support topics

For product and order information:

- Visit www.agilent.com/find/accessories
- Use the search function to locate information about a particular accessory or view the entire RF and Microwave Test Accessories Catalog.

Accessories are available in these connector types:

- 50 ohm Type-N
- 75 ohm Type-N
- 3.5 mm
- 7 mm (APC-7)
- 7-16
- 2.92 mm
- 2.4 mm
- 1.85 mm
- 1 mm

Test port cables and a calibration kit are necessary for a complete measurement system.

A verification kit is used to verify corrected system performance.

[See the connector type for each PNA model](#)

Coax Mechanical Calibration Kits

Model	Connector Type	Frequency Upper Limit
85032B	Type-N (50 Ohm)	6 GHz
85032F	Type-N (50 Ohm)	9 GHz
85054B	Type-N (50 Ohm)	18 GHz
85036E	Type-N (75 Ohm)	3 GHz
85050B	7 mm	18 GHz
85033D	3.5 mm	6 GHz
85038A	7-16	7.5 GHz
85033E	3.5 mm	9 GHz
85052B	3.5 mm	26.5 GHz
85052C	3.5 mm TRL	26.5 GHz
85056K	2.92 mm	50 GHz
85056A	2.4 mm	50 GHz
85058B/E (data-based)	1.85 mm	67 GHz
85059A (data-based)	1.00 mm	110 GHz

Waveguide Mechanical Calibration Kits

Model	Connector Type	Frequency Range
X11644A	WR-90	8.2-12.4 GHz
P11644A	WR-62	12.4-18 GHz
K11644A	WR-42	18-26.5 GHz
R11644A	WR-28	26.5-40 GHz
Q11644A	WR-22	33-50 GHz
U11644A	WR-19	40-60 GHz
V11644A	WR-15	50-75 GHz

Electronic Calibration (ECal)

Model	Connector Type	Frequency Range
RF Two-Port		
85091C	7 mm (APC-7)	300 kHz-9 GHz
85092C	Type-N (50 ohm) Port B available with 3.5 mm or 7-16 ^a	300 kHz-9 GHz
85093C	3.5 mm Port B available with Type-N (50 ohm) or 7-16 ^a	300 kHz-9 GHz
85096C	Type-N (75 ohm)	300 kHz-3 GHz
85098C	7-16 ^a Port B available with Type-N (50 ohm) or 3.5 mm	300 kHz-7.5 GHz
85099C	Type-F	300 kHz-3 GHz
RF Four-Port		
N4431B	3.5mm (f) (four-port), Type-N (f) (four-port), Mixed connector types	9 kHz ^b -13.5 GHz
N4432A Option 020	Type-N (f) (four-port)	300 kHz-18 GHz (available Feb. 2006)
N4432A Option 030	APC 7 (four-port)	300 kHz-18 GHz (available Feb. 2006)

N4433A Option 010	3.5mm (f) (four-port)	300 kHz-20 GHz (available Feb. 2006)
Microwave Two-Port		
N4690B	Type-N (50 ohm)	300 kHz-18 GHz
N4691B	3.5 mm	300 kHz-26.5 GHz
N4692A	2.92 mm	10 MHz-40 GHz
N4693A	2.4 mm	10 MHz-50 GHz
N4694A	1.85 mm	10 MHz-67 GHz
N4696BA	7 mm	300 kHz-18 GHz

a Limits ECal module high frequency to 7.5 GHz.

b Performance from 9 kHz to 300 kHz is valid only for the E5071C ENA Network analyzer with firmware version A.09.10 or higher.

Verification Kits

Model	Connector Type	Frequency Range
85055A	Type-N (50 Ohm)	300 kHz-9 GHz
85053B	3.5 mm	300 kHz-26.5 GHz
85057B	2.4 mm	.045-50 GHz
R11645A	WR-28	26.5-40 GHz
Q11645A	WR-22	33-50 GHz

Adapters and Accessory Kits

Model	Description
11878A	Type-N to 3.5 mm Adapter Kit
11525A	Type-N (m) to 7 mm (APC-7)
11853A	Type-N Accessory Kit
11900B	2.4 mm (f) to 2.4 mm (f)
11900C	2.4 mm (f) to 2.4 mm (m)
85130G	Test Port Adapter Set, 2.4 mm (f) to 2.4 mm (m,f)
11901B	2.4 mm (f) to 3.5 mm (f)
11901D	2.4 mm (f) to 3.5 mm (m)
85130F	Test Port Adapter Set, 2.4 mm (f) to 3.5 mm (m,f)
11902B	2.4 mm (f) to 7 mm (APC-7)
11920A	1 mm (m) to 1 mm (m)
11920B	1 mm (f) to 1 mm (f)
11920C	1 mm (m) to 1 mm (f)
11921A	1 mm (m) to 1.85 mm (m)
11921B	1 mm (f) to 1.85 mm (f)
11921C	1 mm (m) to 1.85 mm (f)
11921D	1 mm (f) to 1.85 mm (m)
11922A	1 mm (m) to 2.4 mm (m)
11922B	1 mm (f) to 2.4 mm (f)
11922C	1 mm (m) to 2.4 mm (f)
11922D	1 mm (f) to 2.4 mm (m)

Test Port Cables

Model	Description
N4697E	1.85 mm (f) to 1.85 mm (rugged f) flexible (single)
N4697F	1.85 mm (rugged f, f) to 1.85 mm (rugged m, rugged f) flexible (set)
N6315A	Type-N (m) to Type-N (f), 16 in. (single)
N6314A	Type-N (m) to Type-N (m), 24 in. (single)
85133D	2.4 mm (f) to 2.4 mm (m,f) semi-rigid (set)
85133F	2.4 mm (f) to 2.4 mm (m,f) flexible (set)
85134D	2.4 mm (f) to 3.5 mm (m,f) semi-rigid (set)
85134F	2.4 mm (f) to 3.5 mm (m,f) flexible (set)

USB Peripherals

Model	Description
N4688A	CD RW drive - with USB cable.
N4689A	USB 4-port hub - for connecting additional USB peripherals.
82357A	USB/GPIB Interface - for controlling GPIB devices through USB. Learn more about using the 82357A with the PNA

Connector and ESD Supplies

[See ESD topic](#)

[See more Connector Care supplies](#)

Part Number	Description
9300-1367	Adjustable antistatic wrist strap
9300-0980	Antistatic wrist strap grounding cord (5 foot)
9300-0797	Static control table mat (2 foot x 4 foot) with earth ground wire
9300-1126	ESD heel strap
1401-0248	ESD Safe End-Cap, Type-N (m)
1401-0247	ESD Safe End-Cap, Type-N (f)
1401-0214	Standard End-Cap, Type-N (m)
1401-0225	Standard End-Cap, Type-N (f)

Last Modified:

26-Feb-2013	Changed Type-N F end cap # and fixed supplies cat
9-Jan-2009	Added footnote for N4431B
1-Dec-2008	Added link to connector care

82357A USB / GPIB Interface

The Agilent 82357A is an adapter that creates a GPIB Interface from one of your unused PNA USB ports.

- [Applications](#)
- [Installing](#)
- [Configuring](#)
- [Connecting](#)
- [Communicating with other Equipment](#)

Applications

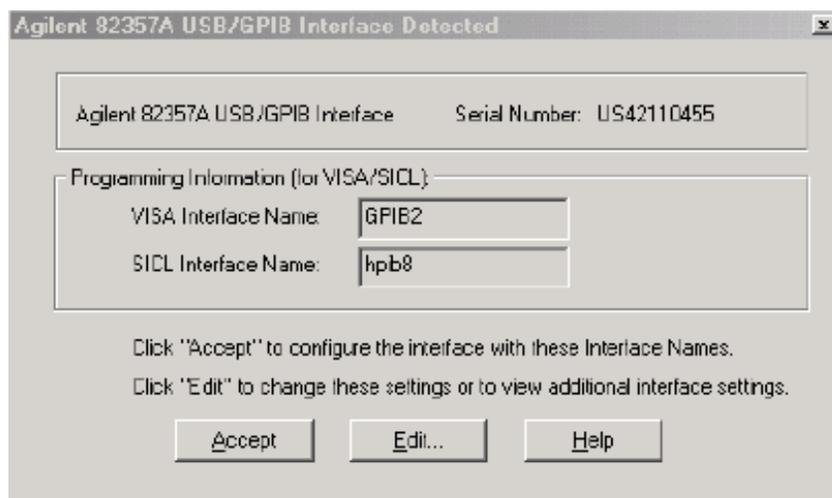
The 82357A can be used to connect a GPIB device using the PNA USB for any PNA application. In addition, the 82357A can be used to connect a power meter for a source power calibration.

Installing the 82357A USB/GPIB Interface

1. [Download and install firmware](#) PNA revision 3.0 or greater. To check the revision of your PNA firmware, click **Help** then **About Network Analyzer**.
2. Upgrade to the latest Agilent IO libraries from the CDROM that was shipped with the 82357A. If not available, download them from www.Agilent.com (search for **82357A**)

Configure the 82357A USB/GPIB Interface

When the 82357A is connected to the PNA USB, the following dialog box appears:



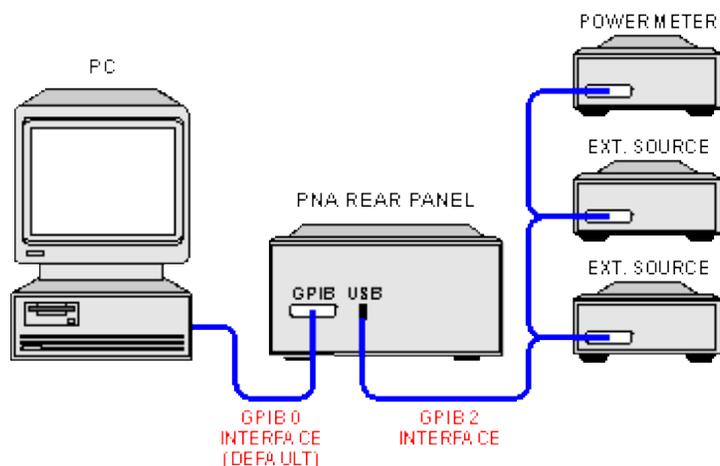
Normally, you do NOT need to edit these settings. The 82357A USB/GPIB Interface is configured automatically as the next unused VISA interface. This is usually **GPIB2** unless you have already configured it for another purpose.

If the VISA Interface Name appears as GPIB0 or GPIB1, these Interfaces must be returned to their default settings for the 82357A to work properly with the PNA. [See Configure for VISA / SICL to learn how.](#)

Connecting the 82357A USB/GPIB Interface

The following diagram illustrates how to connect GPIB test equipment using the USB/GPIB Interface.

- Plug the USB/GPIB Interface into any unused PNA USB port.
- The GPIB Interface and USB/GPIB Interface should never be connected together.



Communicating with Equipment Connected to the USB/GPIB Interface

- The Frequency Converter Application will automatically find and communicate with test equipment that is connected to the USB/GPIB Interface.
- Source power calibration: Select **GPIB** at the [Power Meter Settings dialog](#) and specify the GPIB address of the power meter.
- To control other devices through your own program using the 82357A, you must include the new GPIB Interface number when addressing the devices.

Last modified:

23-Apr-2012 SPC update

Firmware Update

PNA firmware updates are available to you at no cost in a self-extracting Install Shield file. The update includes the PNA application, Online help, and Service Utilities. Note: The file is **at least 50 MB**.

Note: The [CPU speed](#), [amount of RAM](#), and [Operating System](#) in your PNA may limit your ability to update firmware. See http://na.tm.agilent.com/pna/firmware/PNA_support_matrix.doc.

The following options are available for you to update your PNA application:

- [Auto-Check](#) and [AgileUpdate](#) If your PNA is connected to the Internet, these utilities will automatically check for, download, and install, the new firmware and associated files when the PNA application is started. You will be prompted before this occurs.
- [Website Access](#) If your PNA is NOT connected to the Internet, but you have a PC that is, you can download the PNA firmware and associated files to a storage medium.

To manually check the version of firmware on the PNA, click **Help**, then [About Network Analyzer](#).

Tip: Use [Move App to Back](#) to cause the PNA application to move behind this application on the screen.

Note: After a firmware update...

- Custom Cal Kits must be imported. [Learn more](#)
- If a different desktop icon named "Network Analyzer" exists, the shortcut to the PNA application will assume the same icon. Right-click on the desktop, then click **Refresh**.

Other Support Topics

Auto-Check

With Internet access to your PNA, Auto-Check automatically and regularly checks the Internet for new PNA firmware revisions. If a new revision is found, a notification message prompts you to run the [AgileUpdate](#) utility, which then performs the actual download.

Without Internet access to your PNA, Auto-Check provides a reminder prompt at the selected intervals.

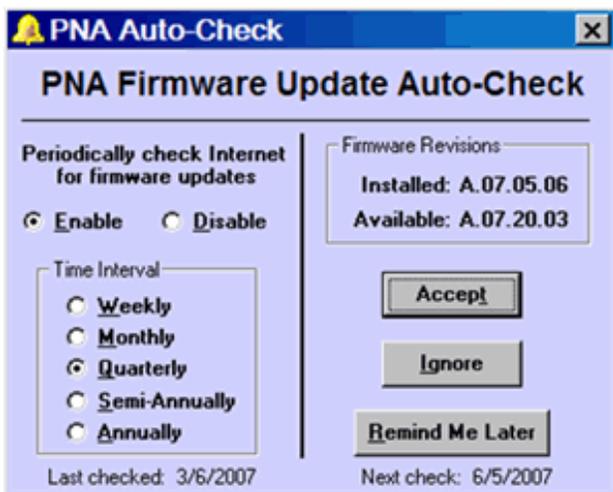
Auto-Check is run only when the PNA application is started. Once the PNA application is running, it will not check for updates again until it is restarted.

When Auto-Check runs, it checks the following conditions:

- Is there an active connection to the Internet?
- Is the Auto-Check utility enabled?
- Is it time to check for new firmware?
- Does new firmware exist?

If all of these conditions are true, Auto-Check shows the following dialog box.

If all of these conditions are NOT true, or to change these settings at any time, press **System**, then **Service**, then **AgileUpdate**. From within AgileUpdate, click **AutoCheck**. These preferences are stored in the PNA registry. Future firmware updates will not change these settings.



PNA Auto-Check dialog box help

Enable When the PNA application is started, Auto-Check will search the PNA website for firmware updates at the selected time interval.

Disable When the PNA application is started, Auto-Check will NOT search the PNA website for firmware updates.

Time Interval Select the time interval Auto-Check is to search for firmware updates.

Accept Starts update process.

Ignore No further action is taken until the selected time interval has elapsed.

Remind Me Later: This window is displayed again after 1-20 days depending upon the time interval selected.

AgileUpdate

Note: You must have administrative privileges on the analyzer to run this utility. See [Set Up Analyzer Users](#).

How to start AgileUpdate

Connect the PNA to the Internet. A LAN connection is recommended because a firmware download can take many hours using a modem.

Using front-panel HARDKEY [softkey] buttons

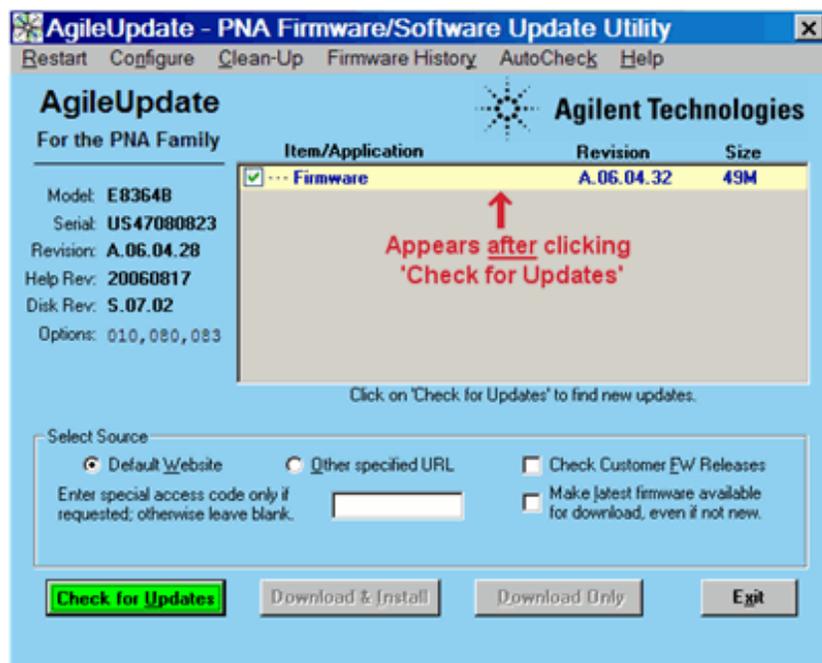
1. Press **SYSTEM**
2. then [**Service**]
3. then [**AgileUpdate**]

Using a mouse with PNA Menus

1. Click Utility
2. then **System**
3. then **Service**
4. then **AgileUpdate**

1. Click **Check for Updates**.
2. If updates exist, click **Download & Install**.

No programming commands are available for this feature



AgileUpdate dialog box help

Note: Your privacy is important to Agilent. AgileUpdate does NOT send ANY information from

the PNA to the server. It only downloads from the server to the PNA.

Restart Click to restart from the beginning.

Configure Click to launch the [Configure dialog box](#).

Clean-up Click to delete all but the two most recent install shield packages from the PNA hard drive.

Firmware History Available after clicking **Check for Updates**.

Auto-Check Launches the [Auto-Check](#) dialog box.

Item / Application Lists the items available for download at the firmware website.

- Click items with **i** to read more information about the download.
- Items in **RED** should be downloaded and installed individually.
- Multi-language help includes all help files except English.

Note: The firmware includes the help file. Therefore, only the firmware checkbox will be selected if a new version for both the firmware and the help file are available.

Select Source

Default Website The Agilent site that contains update FW.

Other Specified URL Click if you were instructed to get firmware from a different website.

Check Customer FW Releases Check this box to also check Customer Releases in addition to Production Releases. This setting provides you with the very latest PNA firmware. Customer Releases are fully supported but have not yet been tested in all production models. Customer Releases take precedence over Production Releases. This setting is remembered and applied the next time AgileUpdate is run.

Special Access Code... Type in the code if you were given one from Agilent Technical Support. Otherwise, leave blank.

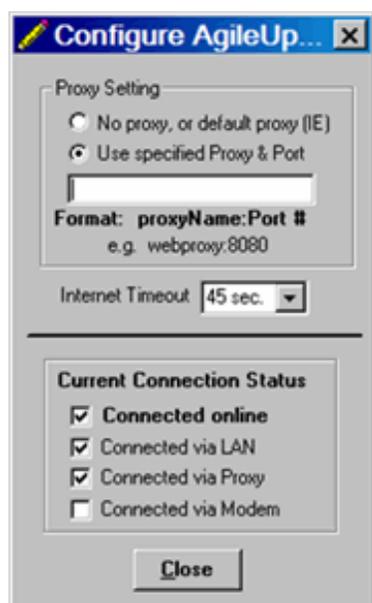
Make Latest Firmware Available... Select this checkbox if you want to download the latest firmware, even if it is not new.

Check for Updates Click to look for firmware updates at the Agilent website. If there are newer versions, the files will be listed.

Download and Install When updates are found, this selection becomes available. Some files may

be pre-checked. Be sure the corresponding boxes are checked for the files you want to download. Then click to download and install the update.

Download Only Click to download the files to the analyzer hard disk and install the files at a later time. At that time, click **Install from File**.



Configuration dialog box help

Note: If AgileUpdate will not connect, try to access ANY Internet website. Contact your local IT department if necessary.

Proxy Setting

No Proxy or Default Proxy Click if you use a LAN connection. AgileUpdate will automatically use the proxy specified in Internet Explorer.

Use specified Proxy / Port Click to enter the proxy name and port. The format is: proxyName:portNumber. (The proxy port number is typically 8088).

Internet timeout If you are using an automatic dial-up Internet connection you may need to increase the timeout.

Current Connection Status Shows the current status of the PNA connection to the Internet.

Note: These settings are NOT saved; they must be re-entered each time AgileUpdate is run.

Agilent Website Access

If you cannot access the Internet directly with your PNA, you can use an external PC with Internet access to download the file from the Agilent website. You can then transfer the file from your PC to your analyzer over a LAN or other means.

1. Go to <http://na.tm.agilent.com/pna/firmware>
2. Click on the firmware to be downloaded.
3. Save the program to disk (hard drive of your PC).
4. Transfer the file from your PC to your PNA using LAN, CD, or USB Pen drive.
5. Double-click the file on the PNA.

Warning: You can save the update file to your PC, but do not attempt to install the PNA application on your PC. It will alter system settings and can result in system crashes.

Last Modified:

29-Aug-2013	Upgrade to update
	Move to back
12-Oct-2009	Added customer release (9.2)
13-Mar-2008	Added limitations note

PNA Configurations and Options

Included with each PNA is a mouse and keyboard. This topic presents PNA models that are supported with this firmware release and the available options.

- [PNA-X Series](#)
- [PNA N522xA Series](#)
- [PNA N523xA \(Updated PNA-L\) Series](#) **New**
- [mmWave Model](#)
- [Measurement Receiver](#) N5264A
- [Common PNA Options](#)
- [Warranty Period](#)

See Also

- [PNA Series Configuration Guide](#) (requires an Internet connection)
- Click **Help** then **About Network Analyzer** to view the options that are installed on your PNA.

Other Support Topics

PNA-X Series

[See Block Diagrams](#)

- **N5241A: 10 MHz to 13.5 GHz**
- **N5242A: 10 MHz to 26.5 GHz**
- **N5244A: 10 MHz to 43.5 GHz**
- **N5245A: 10 MHz to 50.0 GHz**
- **N5247A: 10 MHz to 67 GHz**
- **N5264A: Measurement Receiver** [Learn more](#)

PNA-X Options

[See other options that MAY be offered on the PNA-X models](#)

[See NVNA Brochure for more PNA-X options.](#)

Option	Required Model/Options	Description												
200	None	2-port model base model. Includes six front-panel access loops.												
219	200	To 2-port base model, add: <ul style="list-style-type: none"> • Bias-tees between each source and each test port • Source and receiver attenuators: <table border="1"> <thead> <tr> <th>Model</th> <th>Source Attn</th> <th>Receiver Attn</th> </tr> </thead> <tbody> <tr> <td>N5241A/42A</td> <td>0 to 65 dB in 5 dB steps</td> <td>0 to 35 dB in 5 dB steps</td> </tr> <tr> <td>N5244A/45A</td> <td>0 to 60 dB in 10 dB steps</td> <td>0 to 35 dB in 5 dB steps</td> </tr> <tr> <td>N5247A</td> <td>0 to 50 dB in 10 dB steps</td> <td>0 to 50 dB in 10 dB steps</td> </tr> </tbody> </table>	Model	Source Attn	Receiver Attn	N5241A/42A	0 to 65 dB in 5 dB steps	0 to 35 dB in 5 dB steps	N5244A/45A	0 to 60 dB in 10 dB steps	0 to 35 dB in 5 dB steps	N5247A	0 to 50 dB in 10 dB steps	0 to 50 dB in 10 dB steps
Model	Source Attn	Receiver Attn												
N5241A/42A	0 to 65 dB in 5 dB steps	0 to 35 dB in 5 dB steps												
N5244A/45A	0 to 60 dB in 10 dB steps	0 to 35 dB in 5 dB steps												
N5247A	0 to 50 dB in 10 dB steps	0 to 50 dB in 10 dB steps												
224	200, one of 219 or H85, and 080	To 2-port model, add internal 2nd source, combiner, and mechanical switches.												
400	None	4-port model base model. Includes twelve front-panel access loops.												
419	400 (080 recommended)	To 4-port model, add: <ul style="list-style-type: none"> • Source and receiver attenuators (see table above for values) • Bias-tees between each source and each test port 												
423	400, one of 419 or H85, and 080	To 4-port model, add combiner and mechanical switches.												

PNA N522xA Series

[See Block Diagrams.](#)

[See Specs](#)

- **N5221A: 10 MHz to 13.5 GHz**
- **N5222A: 10 MHz to 26.5 GHz**
- **N5224A: 10 MHz to 43.5 GHz**
- **N5225A: 10 MHz to 50.0 GHz**

- **N5227A: 10 MHz to 67 GHz**

The N522xA models are identical to the PNA-X series **except**:

- N522xA 2-port models are NOT available with 2 sources.
- N522xA option numbering is slightly different.
- N522xA models do NOT have internal **RF switches** or **combiners** (no [RF Path Configuration](#)). This has many measurement implications.
- N522xA models do NOT have [rear-panel access to RF Paths](#).
- N522xA models do NOT offer a Noise Receiver (Opt 029).

PNA N522xA Options

[See other options that MAY be offered on the N522xA models](#)

Option	Required Model/Options	Description												
200	None	2-port model with single RF source.												
201	All Models	To Opt 200, adds six front-panel jumpers and R1 reference receiver switch.												
217	NOT available on N5227A 200 and 080	To Opt 201, adds source and receiver attenuators . <table border="1"> <thead> <tr> <th>Model</th> <th>Source Attn</th> <th>Receiver Attn</th> </tr> </thead> <tbody> <tr> <td>N5221A/22A</td> <td>0 to 65 dB in 5 dB steps</td> <td>0 to 35 dB in 5 dB steps</td> </tr> <tr> <td>N5224A/25A</td> <td>0 to 60 dB in 10 dB steps</td> <td>0 to 35 dB in 5 dB steps</td> </tr> <tr> <td>N5227A</td> <td>0 to 50 dB in 10 dB steps</td> <td>0 to 50 dB in 10 dB steps</td> </tr> </tbody> </table> <p>Note: For N5227A models, the extended power range is available ONLY with Opt 219 and 419.</p>	Model	Source Attn	Receiver Attn	N5221A/22A	0 to 65 dB in 5 dB steps	0 to 35 dB in 5 dB steps	N5224A/25A	0 to 60 dB in 10 dB steps	0 to 35 dB in 5 dB steps	N5227A	0 to 50 dB in 10 dB steps	0 to 50 dB in 10 dB steps
Model	Source Attn	Receiver Attn												
N5221A/22A	0 to 65 dB in 5 dB steps	0 to 35 dB in 5 dB steps												
N5224A/25A	0 to 60 dB in 10 dB steps	0 to 35 dB in 5 dB steps												
N5227A	0 to 50 dB in 10 dB steps	0 to 50 dB in 10 dB steps												
219	200	To Opt 217 adds bias-tees between each source and each test port.												
400	None	4-port model with two sources.												
401	400	To Opt 400, adds 12 front-panel jumpers and R1 reference receiver switch.												

417	400	To Opt 401 adds source and receiver attenuators. See table above for values. Note: For N5227A models, the extended power range is available ONLY with Opt 219 and 419.
419	400 (080 recommended)	To Opt 417 adds bias-tees between each source and each test port.

N523xA (Updated PNA-L) Series New

[See Specs](#)

- **N5231A: 300 kHz to 13.5 GHz** 2-port / 4-port
- **N5232A: 300 kHz to 20.0 GHz** 2-port / 4-port
- **N5234A: 10 MHz to 43.5 GHz** 2-port ONLY
- **N5235A: 10 MHz to 50.0 GHz** 2-port ONLY
- **N5239A: 300 kHz to 8.5 GHz** 2-port ONLY

The N523xA models are identical to the N5230C series **except:**

- N523xA models have a PNA-X look and feel.
- N523xA models are NOT available with 2 sources.
- ALL N5231A, N5232A, and N5239A models require a [Delta Match calibration](#).

N523xA Options

[See other options that MAY be offered on the N523xA models](#)

Option	Required Model/Options	Description
200	None	Base 2-port model.
216	None	To base 2-port model, adds: <ul style="list-style-type: none"> • Six front-panel jumpers • Source Attenuator: 60 dB with 10 dB steps.
400	None	Base 4-port model. Available ONLY on N5231A and N5232A
416	None	To base 4-port model, adds: <ul style="list-style-type: none"> • Six front-panel jumpers • Source Attenuator: 60 dB with 10 dB steps.

Millimeter Wave PNA

PNA Model	Frequency Range	Ports	Connector Type
N5250C	10 MHz to 110 GHz	2	1.0 mm
N5251A	10 MHz to 110 GHz	2	1.0 mm

Test heads to 325 GHz are also available.

Measurement Receiver

Block Diagrams are available at the end of each [specifications](#) document.

PNA Model	Frequency Range	Ports	Connector Type	Reference Receivers	Front Panel jumpers
N5264A	IF Frequencies only	0	N/A	1	N/A

[Learn more](#)

Options

108 - Built-in 26.5 GHz LO source with +10 dBm output power.

118 - Fast-CW mode enables 500 million point data buffer.

Common Options

The following options are common to more than one PNA family.

Option	Available on models:	Description
008	PNA-X N522xA Requires Opt 025	Integrated Pulsed Application Provides average pulse and point-in-pulse measurements. Learn more.
010	ALL models	Time-domain Adds time-domain capability to analyzer. The serial number of the analyzer must be specified when ordering this kit. Software upgrade. Learn more about Time Domain Learn how this option is enabled.
020	PNA-X N522xA	Add IF inputs on the rear panel for antenna and millimeter-wave.
021	PNA-X N522xA	Add pulse modulator to internal Source1. Learn more.
022	PNA-X 2-source N522xA models	Add pulse modulator to internal Source2. Learn more.
025	PNA-X N522xA	Add four internal pulse generators. Learn more.
028	PNA-X and N522xA with 080	Noise Figure measurements on amplifiers or converters using a standard PNA receiver.
029	PNA-X with 080 and one of the following: 219, 224, 419, 423 or H85	Noise Figure Application. Adds hardware and firmware for high-accuracy noise figure measurements on amplifiers or converters using internal low-noise receivers or a standard PNA receiver (Opt 028). For measuring converters, requires Opt 082 or 083.
H29	N5244A or N5245A with 423 and 080.	Same as Opt 029, but on a N5244A or N5245A . Learn more.

080	All models	<p>Frequency Offset Mode (FOM)</p> <p>Enables you to set the PNA source independently from where the receivers are tuned. This capability is important for measuring mixers and converters. Learn more.</p>
082	All models	<p>Scalar Mixer Measurements (SMC)</p> <p>Allows Only the Scalar Mixer Converter (SMC) portion of the Frequency Converter Measurement Application. Provides the same intuitive user-interface, easy calibration, and external source control for making fixed and swept LO Scalar Mixer measurements. When used with a multiport PNA or external test set, SMC is only available on PNA ports 1 and 2.</p> <p>Requires Opt 080.</p> <p>Learn more.</p>
083	PNA-X N522xA	<p>Frequency Converter Application (FCA)</p> <p>Provides an intuitive user-interface for making extremely accurate conversion loss and absolute group delay measurements on mixers and converters. Exceptional amplitude and phase accuracy is achieved through two calibration techniques: Scalar Mixer Calibration and Vector Mixer Calibration. The application also provides automatic control of all of Agilent's major signal sources.</p> <p>Requires Opt 080.</p> <p>Learn more</p>
084	PNA-X N522xA	<p>Embedded LO</p> <p>Provides the ability to measure frequency converters that have an embedded LO. Requires at least one Converter App option.</p> <p>Learn more</p>
086	PNA-X N522xA	<p>Gain Compression Application. (GCA)</p> <p>Provides fast and accurate gain compression measurements.</p> <p>Learn more.</p> <p>For measuring converters, requires Opt 082 or 083.</p>
087	PNA-X N522xA	<p>Swept IMD and IM Spectrum.</p> <p>Provides fast and accurate Swept IMD and IM Spectrum</p>

	(NOT available with Opt 200 or 400)	measurements on amplifiers and converters. Learn more. Requires Opt 080. For measuring converters, requires Opt 082 or 083.
088	PNA-X N522xA (NOT available with 2-port models)	Source Phase Control Provides coherent phase measurements. Learn more.
H85	N5242A	Allows high-power measurements up to 20 Watts (+43 dBm). from 10 MHz to 26.5 GHz. Similar to the PNA-X -219 or -419 but deletes the bias tees from the test set. Learn more.
108	N5264A Only	Adds an internal 10 MHz to 26.5 GHz LO source. Learn more.
118	PNA-X N522xA	Fast CW Mode Enables 500 million point data buffer. Learn more.
301	All models	CalPod (Standard) Allows recalibration from a remote location. Learn more.
302	All models	CalPod (Thermal characterization)
304	All models	CalPod as ECal Use your configured CalPod module to perform 1-port calibrations. Learn more.
460	4-port PNA-X and N522xA	iTMSA Adds firmware for Integrated True Mode Balanced measurements. Learn more.
551	All models	N-port capabilities Adds fully integrated measurements at all of the available test ports using a multiport testset. Learn more. Solutions and PNA requirements depend on the supported test set. See http://www.agilent.com/find/multiport

Certification Options for ALL models

The following options are available for your PNA.

Option	Supported Models	Description
897	All models	Perpetual license for built-in performance test software for Agilent inclusive cal. Adds built-in performance testing and calibration software for self-maintainers. Requires additional equipment. See your PNA Service Guide for more information on equipment required.
898	All models	Perpetual license for built-in performance test software for Standards compliant cal. Adds built-in performance testing and calibration software for self-maintainers. Requires additional equipment. See your PNA Service Guide for more information on equipment required.
UK6	ALL except N5250A	Complete set of measurement data which was acquired from testing your PNA to published specifications. Includes calibration label, calibration certificate, and data report. Conforms to ISO 9001.
1A7	ALL except N5250A	Complete set of measurement data which was acquired from testing your PNA to published specifications. Includes calibration label, ISO 17025 calibration certificate, data report, measurement uncertainties and guard bands on all specifications. Conforms to ISO17025 and ISO 9001.

Documentation

Description
Printed versions of PNA Help in pdf format are available at www.agilent.com/find/pna . (Apr.2005)
A documentation CD-ROM is no longer included with each PNA shipment (Feb.2005).
To download a service guide for your PNA, or the latest version of PNA Help, visit www.agilent.com/find/pna , search for your PNA model, then click Library.

PNA Warranty Period

The actual warranty on your instrument depends on the date it was ordered as well as whether or not any warranty options were purchased at that time. To determine the exact warranty on your instrument, contact [Agilent Technologies](http://www.agilent.com) with the model and serial number of your instrument.

For online information about Agilent's service and support products visit:

www.agilent.com/find/tm_services.

Last modified:

7-Mar-2013	Added CalPod
10-May-2012	Added N523xA models
20-Mar-2012	Many changes driven by no more c models
21-Jun-2011	Updated link to new config guide
29-Apr-2011	Added new N522xA models
2-Nov-2010	Added link to config guide, N5247A, and 088
3-Mar-2010	Added 028 and H29
17-Dec-2009	Corrected Certification Options
26-Oct-2009	Added 029 requirements
30-Apr-2009	Added new PNA-X models and PNA-X w/ Opts 081 and 014
8-Sep-2008	Added IMD
9-Jun-2008	Edits for C models and N5242A
25-Oct-2007	Added C models
5-Sep-2007	Added Embedded LO
9/26/06	MQ Added PNA-L 4port models
9/27/06	MX Added PNA-X

Option Enable Utility

The Option Enable utility allows you to perform the following activities on your PNA.

- Enable or remove software options and some hardware options.
- Recover option data if the hard drive or other data-containing assembly is replaced.
- Input or change a serial number.

The following items are discussed in this topic:

- [Keywords](#)
- [Running the Program](#)
- [Removing an Option](#)
- [Installing an Option](#)
- [Repairing and Recovering Option Data](#)
- [Installing or Changing the Serial Number](#)

See Also

[See PNA Configurations and Options](#)

Tip: Use [Move App to Back](#) to cause the PNA application to move behind this application on the screen.

Keywords

To add certain options, you need a keyword that is provided by Agilent. There are two types of keywords:

- **Option Keywords** add a software option.
- **Model Keywords** may be required if you replace multiple assemblies.

Keywords are linked to the PNA **Host ID**, which is displayed on the Option Enable dialog box (below).

Temporary and Permanent Options

Any software option can also be installed on a temporary basis for a specified amount of time. This

allows you to evaluate a specific feature or capability at no cost.

If the license key provided by Agilent has an expiration date, you must select the "temporary" option and enter the expiration date exactly as stated in the license statement. If you decide to make this option permanent, Agilent will provide a new keyword that converts the option to permanent status.

For either permanent or temporary software options, a provided keyword must be entered.

Running the Program

On the PNA, press **System**, then **Service**, then **Option Enable**.

1. To enable or remove an option, select it from the drop-down list of available options. If the desired option is not listed, select the last choice in the list, labeled **Enter Unlisted Option**.
2. Enter the 3-character option name and click **Enter**

If a software option was chosen, the following occurs.

- The **Remove** button will be enabled.
- The keyword entry area becomes visible.
- The permanent/temporary selection is enabled.

If a hardware option is selected, the following occurs.

- With the hardware option already installed, the **Remove** button is enabled.
- With the hardware option not installed, the **Enable** button is enabled.

Removing an Option

1. To remove an option, click **Remove**.
2. After the option is removed, restart the network analyzer application for the changes to take effect.

Note: Removal of a licensed option (such as Option 010, Time Domain) will permanently remove the license keyword. If this option **may** be needed in the future, then record the license keyword before removing the option. Do this by copying the file “gen.lic” to another location (such as a floppy disk), or print it using notepad. The file, located at "c:\users\public\network analyzer\" contains all the information needed to recreate the license.

Installing an Option

1. If the keyword entry area is visible, enter a keyword. (The keyword is not case sensitive.)
2. Click **Enable**.
3. After the option is installed, restart the network analyzer application for the changes to take effect.

Note: If a desired option is not visible, it may be because a prerequisite option has not yet been installed. For example, Option 083 will not be visible if Option 080 is not already present. [See PNA options.](#)

Repairing and Recovering Option Data

Use this part of the Option Enable Utility in the following situations:

- If the hard drive is replaced
- If the frequency reference assembly is replaced

This routine rebuilds the option information contained on the hard drive and frequency reference assembly (primary and backup).

1. Select **Repair** from the **Option Enable** menu bar.

Note: If you are unsure if this routine needs to be done, run it; no harm will result.

-
2. The model and serial number are displayed, along with four check boxes.
 3. Select the boxes that apply.
 4. Click **Begin Repair**. The routine checks all data files and performs any needed repairs. You may be asked to verify certain information and processes.
 5. If the routine finds that the model number is incorrect or invalid, you will be asked to select the correct model number.
 - Along with this model number, a model keyword will be required. If this is not labeled on the analyzer, or is not otherwise known, contact Agilent
 - After you have entered the requested data, click **Change Model**. This process takes about 30 seconds.
 6. When done, click **Exit Repair**.
 7. If you do not need to install any other options, click **Exit**.

Installing or Changing the Serial Number

It may be necessary to install or change a serial number if certain assemblies are replaced.

1. To change the serial number, select **Change Serial** from the **Option Enable** menu bar. The current serial number will be displayed. If no serial number has previously been entered, the word "NONE" will be displayed.
2. Type the new serial number into the space provided, and click **Change Serial**. (The serial number is not case sensitive.)

Note: Use extreme care when entering the serial number; only one entry chance is allowed!

3. To change an incorrect serial number, a clear-code password is required. Contact Agilent to obtain this clear code and have the existing serial number available. Enter the clear code in the space provided, along with the new serial number, then click **Change Serial**.

Last Modified:

20-Sep-2007 Added Install note

Instrument Calibration

An instrument calibration is a process where the PNA performance is measured to ensure that it operates within specifications. If any performance parameter does not conform to the published specifications, adjustments are made to bring the performance into conformance.

Why Should I Get an Instrument Calibrated?

Over time, the active components in the analyzer age and the performance may degrade or drift.

To ensure that the analyzer is performing to the published specifications, you should have an instrument calibration performed periodically.

How Often Should I Get an Instrument Calibrated?

It is your responsibility to determine the calibration period which best meets your requirements. However, a 12 to 18 month calibration cycle is appropriate for most users.

There are two things to consider: performance drift and connector wear.

- The instrument specifications are set to consider the performance drift that may occur over a 24 month period. Therefore, getting the instrument calibrated at 24 month intervals ensures that the analyzer maintains performance within the operating specifications. If you need the analyzer to maintain more consistent operation, you may want to have the instrument calibrated more often than the recommended 24-month interval.
- Connector wear is a bigger factor and depends on the number of connections that are made. The test ports become noticeably worn after 500 to 700 connections. This could represent about 12 months with average use. With more frequent connections, the calibration cycle should be sooner. You can extend the time between calibrations and thereby save money by using [connector savers](#) and by performing proper [Connector Care](#).

How Do I Get an Instrument Calibrated?

To get the instrument calibrated, send it to one of the Agilent Technologies service centers. See [Technical Support](#).

The PNA must be fully functional when it is sent to the service center, or they will charge for their repair services. If the PNA is being used in a secure environment where the hard drive can not be sent with the PNA, a second hard drive must be purchased and configured for use with the PNA in an "unclassified" environment before the PNA is sent to the service center.

To perform the instrument calibration yourself, you must have the following required items:

- Instrument Calibration Test Equipment

- Performance Test Software

What Are My Choices of Instrument Calibration?

The following types of instrument calibration are available from Agilent Technologies at the time of initial order:

- | | |
|-------------------|--|
| Standard | Includes a certificate of calibration stating the instrument has been calibrated and is operating within the published specifications. |
| Option UK6 | Available ONLY at the initial shipment. Includes the test data from the calibration and the certificate of calibration stating the instrument has been calibrated and is operating within the published specifications. |
| Option A6J | Available ONLY at the initial shipment. Includes the test data and measurement uncertainties from the calibration and the certificate of calibration stating the instrument has been calibrated using a process in compliance with ANSI Z540 and is operating within the published specifications. |
| Option 1A7 | Available ONLY at the initial shipment. Includes the test data and measurement uncertainties from the calibration and the certificate of calibration stating the instrument has been calibrated using a process in compliance with ISO 17025 and is operating within the published specifications. |

The following types of instrument calibration are available from Agilent Technologies service center:

- | | |
|------------------------------|---|
| Agilent Calibration | Includes the test data from the calibration and the certificate of calibration, stating the instrument has been calibrated and is operating within the published specifications. |
| ANSI Z540 Calibration | Includes the test data from the calibration and the certificate of calibration, stating the instrument has been calibrated using a process in compliance with ANSI Z540.1 and is operating within the published specifications. |
| ISO 17025 Calibration | Includes the test data from the calibration and the certificate of calibration, stating the instrument has been calibrated using a process in compliance with ISO 17025 and is operating within the published specifications. |

For more information on these options, visit www.agilent.com/find/calibration.

Last Modified:

24-Mar-2010 Updated options and cal cycle

Other Resources

The following network analysis resources are also available.

Document Resources

Application Notes

You can also access application notes at this URL:

<http://www.agilent.com/find/PNA>

Third-Party Resources

For information about test fixtures and part handlers, contact:

Inter-Continental Microwave

www.icmicrowave.com

For information about probing equipment and accessories, contact:

Cascade Microtech, Inc.

www.cascademicrotech.com

SCPI Errors

SCPI Errors

- [-100 to -200 Command Errors](#)
- [-200 to -299 Execution Errors](#)
- [-300 to -399 SCPI Specified Device-Specific Errors](#)
- [-400 to -800 Query and System Errors](#)
- [100 to 200 PNA-specific Errors](#)

See Also

[PNA Error messages.](#)

-100 to -200 Command Errors

A command error indicates that the test set's GPIB parser has detected an IEEE 488.2 syntax error. When one of these errors is generated, the command error bit in the event status register is set.

-100 std_command	Command - This event bit (Bit 5) indicates a syntax error, or a semantic error, or a GET command was entered, see IEEE 488.2, 11.5.1.1.4.
-101 std_invalidChar	Invalid character - Indicates a syntactic elements contains a character which is invalid for that type.
-102 std_syntax	Syntax - Indicates that an unrecognized command or data type was encountered. For example, a string was received when the device does not accept strings.
-103 std_invalidSeparator	Invalid separator - The parser was expecting a separator and encountered an illegal character. For example, the semicolon was omitted after a program message unit.
-104 std_wrongParamType	Data type -The parser recognized a data element different than one allowed. For example, numeric or string data was expected but block data was encountered.
-105 std_GETNotAllowed	GET not allowed - Indicates a Group Execute Trigger was received within a program message. Correct the program so that the GET does not occur within the program code.
-108 std_tooManyParameters	Parameter not allowed - Indicates that more parameters were received than expected for the header. For example, *ESE common command only accepts one parameter, so *ESE 0,1 is not allowed.

-109 std_tooFewParameters	Missing parameter - Indicates that less parameters were received than required for the header. For example, *ESE requires one parameter, *ESE is not allowed.
-110 std_cmdHeader	Command header - Indicates an error was detected in the header. This error is used when the device cannot detect the more specific errors -111 through -119.
-111 std_headerSeparator	Header separator - Indicates that a character that is not a legal header separator was encountered while parsing the header.
-112 std_IDTooLong	Program mnemonic too long - Indicates that the header contains more than twelve characters, see IEEE 488.2, 7.6.1.4.1.
-113 std_undefinedHeader	Undefined header - Indicates the header is syntactically correct, but it is undefined for this specific device. For example, *XYZ is not defined for any device.
-114 std_suffixOutOfRange	Header suffix out of range - Indicates the value of a header suffix attached to a program mnemonic makes the header invalid.
-120 std_numericData	Numeric data - This error, as well as errors
-121 std_invalidCharInNumber	Invalid character in number - Indicates an invalid character for the data type being parsed was encountered. For example, an alpha in a decimal numeric or a "9" in octal data.
-123 std_exponentTooLarge	Exponent too large - Indicates the magnitude of an exponent was greater than 32000, see IEEE 488.2, 7.7.2.4.1.
-124 std_decimalTooLong	Too many digits - Indicates the mantissa of a decimal numeric data element contained more than 255 digits excluding leading zeros, see IEEE 488.2, 7.7.2.4.1.
-128 std_numericNotAllowed	Numeric data not allowed - Indicates that a legal numeric data element was received, but the device does not accept one in this position for the header.
-130 std_suffix	Suffix - This error, as well as errors -131 through -139, are generated when parsing a suffix. This particular error message is used if the device cannot detect a more specific error.
-131 std_badSuffix	Invalid suffix - Indicates the suffix does not follow the syntax described in IEEE 488.2, 7.7.3.2, or the suffix is inappropriate for this device.
-134 std_suffixTooLong	Suffix too long - Indicates the suffix contain more than 12 characters, see IEEE 488.2, 7.7.3.4.
-138 std_suffixNotAllowed	Suffix not allowed - Indicates that a suffix was encountered after a numeric element that does not allow suffixes.
-140 std_charData	Character data - This error, as well as errors

-141 std_invalidCharData	Invalid character data - Indicates that the character data element contains an invalid character or the particular element received is not valid for the header.
-144 std_charDataTooLong	Character data too long - Indicates the character data element contains more than twelve characters, see IEEE 488.2, 7.7.1.4.
-148 std_charNotAllowed	Character data not allowed - Indicates a legal character data element was encountered where prohibited by the device.
-150 std_stringData	String data - This error, as well as errors
-151 std_stringInvalid	Invalid string data - Indicates that a string data element was expected, but was invalid, see IEEE 488.2, 7.7.5.2. For example, an END message was received before the terminal quote character.
-158 std_stringNotAllowed	String data not allowed - Indicates that a string data element was encountered but was not allowed by the device at this point in parsing.
-160 std_blockData	Block data - This error, as well as errors -161 through -169, are generated when parsing a block data element. This particular error message is used if the device cannot detect a more specific error.
-161 std_badBlock	Invalid block data - Indicates a block data element was expected, but was invalid, see IEEE 488.2, 7.7.6.2. For example, an END message was received before the end length was satisfied.
-168 std_blockNotAllowed	Block data not allowed - Indicates a legal block data element was encountered, but not allowed by the device at this point in parsing.
-170 std_expr	Expression - This error, as well as errors -171 through -179, are generated when parsing an expression data element. This particular error message is used if the device cannot detect a more specific error.
-171 std_invalidExpression	Invalid expression - Indicates the expression data element was invalid, see IEEE 488.2, 7.7.7.2. For example, unmatched parentheses or an illegal character.
-178 std_exprNotAllowed	Expression data not allowed - Indicates a legal expression data was encountered, but was not allowed by the device at this point in parsing.
-180 std_macro	Macro - This error, as well as error -181 through -189, are generated when defining a macro or execution a macro. This particular error message is used if the device cannot detect a more specific error.

- 181 std_validOnlyInsideMacro Invalid outside macro definition - Indicates that a macro parameter place holder was encountered outside of a macro definition.
- 183 std_invalidWithinMacro Invalid inside macro definition - Indicates that the program message unit sequence, sent with a *DDT or a *DMC command, is syntactically invalid, see IEEE 488.2, 10.7.6.3.
- 184 std_macroParm Macro parameter - Indicates that a command inside the macro definition had the wrong number or type of parameters.

-200 to -299 Execution Errors

These errors are generated when something occurs that is incorrect in the current state of the instrument. These errors may be generated by a user action from either the remote or the manual user interface

- 200 std_execGen Execution - This event bit (Bit 4) indicates a PROGRAM DATA element following a header was outside the legal input range or otherwise inconsistent with the device's capabilities, see IEEE 488.2, 11.5.1.1.5.
- 201 std_invalidWhileInLocal Invalid while in local
- 202 std_settingsLost Settings lost due to rtl
- 203 std_commandProtected Command protected - Indicates that a legal password-protected program command or query could not be executed because the command was disabled.
- 210 std_trigger Trigger
- 211 std_triggerIgnored Trigger ignored
- 212 std_armIgnored Arm ignored
- 213 std_initIgnored Init ignored
- 214 std_triggerDeadlock Trigger deadlock
- 215 std_armDeadlock Arm deadlock
- 220 std_parm Parameter - Indicates that a program data element related error occurred.
- 221 std_settingsConflict Settings conflict - Indicates that a legal program data element was parsed but could not be executed due to the current device state.
- 222 std_dataOutOfRange Data out of range - Indicates that a legal program data element was parsed but could not be executed because the interpreted value was outside the legal range defined by the devices

-223	std_tooMuchData	Too much data - Indicates that a legal program data element of block, expression, or string type was received that contained more data than the device could handle due to memory or related device-specific requirements.
-224	std_illegalParmValue	Illegal parameter value - Indicates that the value selected was not part of the list of values given.
-225	std_noMemoryForOp	Out of memory - The device has insufficient memory to perform the requested operation.
-226	std_listLength	Lists not same length - Attempted to use LIST structure having individual LIST's of unequal lengths.
-230	std_dataCorruptOrStale	Data corrupt or stale - Indicates invalid data, a new reading started but not completed since the last access.
-231	std_dataQuestionable	Data questionable - Indicates that measurement accuracy is suspect.
-232	std_invalidFormat	Invalid format
-233	std_invalidVersion	Invalid version - Indicates that a legal program data element was parsed but could not be executed because the version of the data is incorrect to the device. For example, a not supported file version, a not supported instrument version.
-240	std_hardware	Hardware - Indicates that a legal program command or query could not be executed because of a hardware problem in the device.
-241	std_hardwareMissing	Hardware missing - Indicates that a legal program command or query could not be executed because of missing device hardware. For example, an option was not installed.
-250	std_massStorage	Mass storage - Indicates that a mass storage error occurred. The device cannot detect the more specific errors described for errors -251 through -259.
-251	std_missingMassStorage	Missing mass storage - Indicates that a legal program command or query could not be executed because of missing mass storage.
-252	std_missingMedia	Missing media - Indicates that a legal program command or query could not be executed because of missing media. For example, no disk.
-253	std_corruptMedia	Corrupt media - Indicates that a legal program command or query could not be executed because of corrupt media. For example, bad disk or wrong format.
-254	std_mediaFull	Media full- Indicates that a legal program command or query could not be executed because the media is full. For example, there is no room left on the disk.
-255	std_directoryFull	Directory full - Indicates that a legal program command or query could not be executed because the media directory was full.

-256	std_fileNotFound	File name not found - Indicates that a legal program command or query could not be executed because the file name was not found on the media.
-257	std_fileName	File name - Indicates that a legal program command or query could not be executed because the file name on the device media was in error. For example, an attempt was made to read or copy a nonexistent file.
-258	std_mediaProtected	Media protected - Indicates that a legal program command or query could not be executed because the media was protected. For example, the write-protect switch on a memory card was set.
-260	std_expression	Expression
-261	std_math	Math in expression
-270	std_macroExecution	Macro - Indicates that a macro related execution error occurred.
-271	std_macroSyntax	Macro syntax - Indicates that a syntactically legal macro program data sequence, according to IEEE 488.2, 10.7.2, could not be executed due to a syntax error within the macro definition.
-272	std_macroExec	Macro execution - Indicates that a syntactically legal macro program data sequence could not be executed due to some error in the macro definition, see IEEE 488.2, 10.7.6.3.
-273	std_badMacroName	Illegal macro label - Indicates that the macro label was not accepted, it did not agree with the definition in IEEE 488.2, 10.7.3
-274	std_macroPlaceholder Ma	cro parameter - Indicates that the macro definition improperly used a macro parameter placeholder, see IEEE 488.2, 10.7.3.
-275	std_macroTooLong	Macro definition too long - Indicates that a syntactically legal macro program data sequence could not be executed because the string of block contents were too long for the device to handle, IEEE 488.2, 10.7.6.1.
-276	std_macroRecursion	Macro recursion - Indicates that a syntactically legal macro program data sequence could not be executed because it would be recursive, see IEEE 488.2, 10.7.6.6.
-277	std_cantRedefineMacro	Macro redefinition not allowed - Indicates that redefining an existing macro label, see IEEE 488.2, 10.7.6.4.
-278	std_macroNotFound	Macro header not found - Indicates that a legal macro label in the *GMS?, see IEEE 488.2, 10.13, could not be executed because the header was not previously defined.
-280	std_program	Program
-281	std_cantCreateProgram	Cannot create program
-282	std_illegalProgramName	Illegal program name
-283	std_illegalVarName	Illegal variable name
-284	std_programRunning	Program currently running

-285	std_programSyntax	Program syntax
-286	std_programRuntime	Program runtime
-290	std_memoryUse	Memory use
-291	std_execOutOfMemory	Out of memory
-292	std_nameNotFound	Referenced name does not exist
-293	std_nameAlreadyExists	Referenced name already exists
-294	std_incompatibleType	Incompatible type

-300 to -399 SCPI Specified Device-Specific Errors

A device-specific error indicates that the instrument has detected an error that occurred because some operations did not properly complete, possibly due to an abnormal hardware or firmware condition. For example, an attempt by the user to set an out of range value will generate a device specific error. When one of these errors is generated, the device specific error bit in the event status register is set.

-300	std_deviceSpecific	Device specific - This event bit (Bit 3) indicates that a device operation did not properly complete due to some condition, such as overrange see IEEE 488.2, 11.5.1.1.6.
-310	std_system	System
-311	std_memory	Memory - Indicates some physical fault in the devices memory, such as a parity error.
-312	std_PUDmemoryLost	PUD memory lost - Indicates protected user data saved by the *PUD command has been lost, see IEEE 488.2, 10.27.
-313	std_calMemoryLost	Calibration memory lost - Indicates that nonvolatile calibration data used by the *CAL? command has been lost, see IEEE 488.2, 10.2.
-314	std_savRclMemoryLost	Save/recall memory lost - Indicates that the nonvolatile data saved by the *SAV command has been lost, see IEEE 488.2, 10.33.
-315	std_configMemoryLost	Configuration memory lost - Indicates that nonvolatile configuration data saved by the device has been lost.
-320	std_storageFault	Storage fault - Indicates that the firmware detected a fault when using data storage. This is not an indication of physical damage or failure of any mass storage element.
-321	std_outOfMemory	Out of memory - An internal operation needed more memory than was available
-330	std_selfTestFailed	Self-test failed - Indicates a problem with the device that is not covered by a specific error message. The device may require service.
-340	std_calFailed	Calibration failed - Indicates a problem during calibration of the device that is not covered by a specific error.
-350	std_queueOverflow	Queue overflow - Indicates that there is no room in the queue and an error occurred but was not recorded. This code is entered into the queue in lieu of the code that caused the error.

-360 std_comm	Communication - This is the generic communication error for devices that cannot detect the more specific errors described for error -361 through -363.
-361 std_parity	Parity in program message - Parity bit not correct when data received for example, on a serial port.
-362 std_framing	Framing in program message - A stop bit was not detected when data was received for example, on a serial port (for example, a baud rate mismatch).
-363 std_inputBufferOverflow	Input buffer overrun - Software or hardware input buffer on serial port overflows with data caused by improper or nonexistent pacing.

-400 to -800 Query and System Errors

A Query error is generated either when data in the instrument's GPIB output queue has been lost, or when an attempt is being made to read data from the output queue when no output is present or pending.

-400 std_queryGen	Query - This event bit (Bit 2) indicates that an attempt to read data from the Output Queues when no output is present or pending, to data in the Output Queue has been lost see IEEE488.2, 11.5.1.1.7.
-410 std_interrupted	Query INTERRUPTED - Indicates the test set has been interrupted by a new program message before it finishes sending a RESPONSE MESSAGE see IEEE 488.2, 6.3.2.3.
-420 std_unterminated	Query UNTERMINATED - Indicates an incomplete Query in the program see IEEE 488.2, 6.3.2.2.
-430 std_deadlocked	Query DEADLOCKED - Indicates that the Input Buffer and Output Queue are full see IEEE 488.2, 6.3.1.7.
-440 std_responseNotAllowed	Query UNTERMINATED after indefinite response - Indicates that a query was received in the same program message after a query requesting an indefinite response was executed see IEEE 488.2, 6.5.7.5.
-500 std_powerOn	Power on
-600 std_userRequest	User request
-700 std_requestControl	Request control
-800 std_operationComplete	Operation complete

PNA Specific (Positive) SCPI Errors

100 dupWindNum	"Duplicate window number"
101 windNumNotFound	"Window number not found"
102 failedWindCreate	"Window creation failed"

103 noCalcParamSelection	"CALC measurement selection set to none"
	See CALC:PAR:SEL
104 dupMeasName	"Duplicate measurement name"
105 dataNotFound	"Requested data not available"
106 measNotFound	"Requested measurement not found"
107 traceNotFound	"Requested trace not found"
108 notImplemented	"Mnemonic not yet implemented"
109 noDocument	"No measurement container found"
110 dupTraceNum	"Duplicate trace number"
111 titleStrTooLong	"Title string exceeds 50 characters"
112 memoryNotFound	"Requested memory not found"
113 exceedMaxTraces	"Exceeded the maximum number of traces per window"
114 SerNumNotFound	"The serial number was not found. Please store the serial number."
115 LoadFailed	"The state was not loaded. Please check the file name."
116 StoreFailed	"The state was not stored. Please check the file and path names."
117 File	"An in the File operation occurred. Please check file and path names."
118 measChanConflict	"Measurement does not belong to specified channel."
119 exceedMaxWindows	"Exceeded the maximum number of data windows"
120 markerNotFound	"The specified marker was not found."
121 diagnostic	"Diagnostic ."
122 channelNotFound	"The specified channel was not found."
123 exceedMaxMeasurements	"Exceeded the maximum number of allowed measurements."
124 parameterOutOfRange	"The specified value was out of range."
125 userRangeNotValid	"The currently selected user range is not valid."
126 referenceMarkerNotFound	"The reference marker is not active."
127 sweepSegmentNotFound	"The sweep segment was not found."
128 markerNotDelta	"The specified marker is not a delta marker."
129 printoutFailed	"Attempt to output to a printer failed."
130 memory_trace_not_compatible	"Memory not compatible. Trace Math not applied."
131 trace_math_reset	"Memory not compatible. Trace Math turned off."
132 hw_read_failed	"Hardware read failed."
133 hw_write_failed	"Hardware write failed."
134 dsp_active	"Failed because DSP was not halted."
135 secure_memory	"Attempt to access secure memory region."
136 snum_protected	"The serial number is protected."

137	snun_format_bad	"The serial number format is bad."
138	snun_already_set	"The serial number is already set."
139	hw_setting_failed	"Hardware setting failed."
140	cal_access_failed	"Calibration data access failed."
141	db_access_failed	"Database access failed."
142	memory_range_exceeded	"Command exceeds usable memory range."
143	lost_phase_lock	"Phase lock has been lost."
144	over_power	"Detected too much power at input."
145	ee_wrt_failed	"EEPROM write failed."
146	yig_cal_failed	"YTO calibration failed."
147	ramp_cal_failed	"Analog ramp calibration failed."
148	dspcom_bad	"DSP communication failed."
149	no_license_found	"Request failed. The required license was not found."
150	argLimited	"The argument was out of range"
151	markerBWNotFound	"The Marker Bandwidth was not found."
153	peakNotFound	"The Peak was not found."
154	targetNotFound	"The Target search value was not found."
155	calNotImpl	"The Calibration feature requested is not implemented."
156	calClassNotValidForCalType	"SENS:CORR:CCH measurement selection set to none"
158	calNotValidForConfidenceChe	"Selected measurement does not have a calibration valid for Confidence Check"
159	invalidPort	"Specified port is out of range"
160	invalidPortPath	"ROUT:PATH:DEF:PORT x, y does not match measurement; setting to defaults"
161	ioInvalidWrite	"Attempted I/O write while port set to read only."
162	ioInvalidRead	"Attempted I/O read from write only port."
163	calsetNotFound	"Requested Cal Set was not found in Cal Set Storage."
164	noCalSetSelected	"There is no Cal Set currently selected for the specified channel."
165	cantDeleteCalSetInUse	"Cannot delete a Cal Set while it is being used."
166	calsetStimChange	"Channel stimulus settings changed to match selected Cal Set."
167	exceedMaxCalSets	"Exceeded the maximum number of cal sets."
168	calCouldNotTurnOn	"A valid calibration is required before correction can be turned on."
169	standardMeasurementRequired	"The attempted operation can only be performed on a standard measurement type."
170	noDivisorBuffer	"A valid divisor buffer is required before normalization can be turned on."

171 InvalidReceiverPowerCalParagraph	"Receiver power cal requires the measurement to be of unratiod power."
172 ecalCouldNotConfigure	"Could not configure the Electronic Calibration system. Check to see if the module is plugged into the proper connector."
173 measHasNoMemoryAlg	"This measurement does not support memory operations"
174 measHasNoNormalizeAlg	"This measurement does not support normalize operations."
175 userCharacterizationNotFound	"User characterization was not found in the Electronic Calibration module."
176 measInvalidBufferSize	"The data provided has an invalid number of points. It could not be stored."

Last Modified:

4-Aug-2009 Cosmetic mods

Technical Support

Click on the region of interest.



For more contact information, visit <http://www.agilent.com/find/contactus>

Other Support Topics

United States:

(tel) (+1) 800-829-4444

(alt) (+1) 303 662 3999

(fax) (+1) 888 900 8921

Canada

(tel) 1 877 894 4414

(fax) 1 (905) 206 4120

Europe:

Austria

(tel) 0820 87 44 11*

(fax) 0820 87 44 22

Belgium

(tel) (+32) (0)2 404 9340

(alt) (+32) (0)2 404 9000

(fax) (+32) (0)2 404 9395

Denmark

(tel) (+45) 7013 1515

(alt) (+45) 7013 7313

(fax) (+45) 7013 1555

Finland

(tel) 08 0052 4000

(alt) (+358) 10 855 2100

(fax) (+358) 92 536 0176

France

(tel) 0825 010 700*

(alt) (+33) (0)1 6453 5623

(fax) 0825 010 701*

Germany

(tel) 01805 24 6333*

(alt) 01805 24 6330*

(fax) 01805 24 6336*

Ireland

(tel) (+353) (0)1 890 924 204

(alt) (+353) (0)1 890 924 206

(fax) (+353) (0)1 890 924 024

Israel

(tel) (+972) 3 9288 500

(fax) (+972) 3 9288 501

Italy

(tel) (+39) (0)2 9260 8484

(fax) (+39) (0)2 9544 1175

Luxemburg

(tel) (+32) (0)2 404 9340

(alt) (+32) (0)2 404 9000

(fax) (+32) (0)2 404 9395

Netherlands

(tel) (+31) (0)20 547 2111

(alt) (+31) (0)20 547 2000

(fax) (+31) (0)20 547 2190

Russia

(tel) (+7) 095 797 3963

(alt) (+7) 095 797 3900

(fax) (+7) 095 797 3901

Spain

(tel) (+34) 91 631 3300

(alt) (+34) 91 631 3000

(fax) (+34) 91 631 3301

Sweden

(tel) 0200 88 22 55*

(alt) (+46) (0)8 5064 8686

(fax) 020 120 2266*

Switzerland (French)

(tel) 0800 80 5353 opt. 2*

(alt) (+33) (0)1 6453 5623

(fax) (+41) (0)22 567 5313

Switzerland (German)

(tel) 0800 80 5353 opt. 1*

(alt) (+49) (0)7031 464 6333

(fax) (+41) (0)1 272 7373

Switzerland (Italian)

(tel) 0800 80 5353 opt. 3*

(alt) (+39) (0)2 9260 8484

(fax) (+41) (0)22 567 5314

United Kingdom

(tel) (+44) (0)7004 666666

(alt) (+44) (0)7004 123123

(fax) (+44) (0)7004 444555

Japan:

(tel) 0120 421 345

(alt) (+81) 426 56 7832

(fax) 0120 421 678

Latin America:

Mexico

(tel) (+52) 55 5081 9469

(alt) 01800 5064 800

(fax) (+52) 55 5081 9467

Brazil

(tel) (+55) 11 4197 3600

(fax) (+55) 11 4197 3800

Australia:

((tel) 1800 629 485

(alt) 1800 143 243

(fax) 1800 142 134

New Zealand

(tel) 0 800 738 378

(fax) 64 4 495 8950

Asia Pacific:

China

(tel) 800 810 0189

(alt) (+86) 10800 650 0021

(fax) 800 820 2816

Hong Kong

(tel) 800 930 871

(alt) (+852) 3197 7889

(fax) (+852) 2 506 9233

India

(tel) 1600 112 929

(fax) 000800 650 1101

Malaysia

(tel) 1800 888 848

(alt) 1800 828 848

(fax) 1800 801 664

Singapore

(tel) 1800 375 8100

(fax) (+65) 6836 0252

South Korea

(tel) 080 769 0800

(alt) (+82) 2 2004 5004

(fax) (+82) 2 2004 5115

Taiwan

(tel) 0800 047 866

(alt) 00801 651 317

(fax) 0800 286 331

Thailand

(tel) 1800 226 008

(alt) (+66) 2 268 1345

(fax) (+66) 2 661 3714

Last Modified:

1-Dec-2008 Changed US number

10 MHz Reference Frequency Adjustment

This routine adjusts the analyzer's internal time-base to exactly 10 MHz by changing a DAC value. This DAC value is stored in the analyzer's non-volatile memory. This routine should only be necessary in the following situations:

- The frequency reference assembly is replaced.
- The 10 MHz reference has drifted significantly from the factory adjusted value.

WARNING: The range of this adjustment is only about 20 Hz. It is highly recommended that a very accurate frequency standard be used to measure this 10 MHz signal.

Frequency Counter Compatibility

This procedure uses SCPI commands (over GPIB) to communicate with the frequency counter. It should work with the Agilent 5313xA, 5315xA, 53181A series of counters as well as the older 5350 series.

If no compatible counters are available, select the "Manual" mode of operation.

Procedures

Note: You must be logged onto the PNA as an Administrator to perform an adjustment. [Learn more.](#)

Click **Utilities**, then **System**, then **Service**, then **Adjustments**

At the Adjustments selection, click **10 MHz Frequency Adjustment**

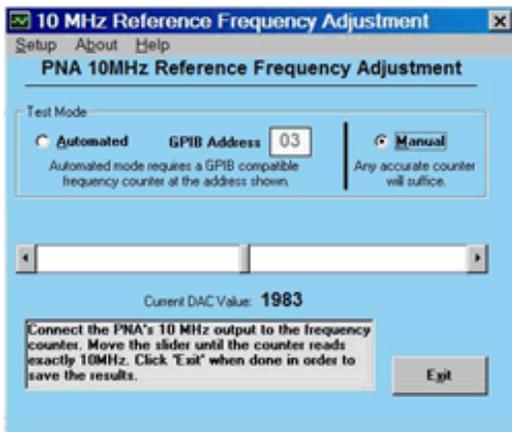
Procedure for GPIB Counters Only

1. Connect the analyzer rear panel 10 MHz Reference output to the frequency counter .
2. Connect a GPIB cable from the analyzer to the counter. Make sure no other controllers are active on the same connection.
3. If applicable, connect the house frequency standard to the counter reference input.
4. Set the counter GPIB address to 03. Ensure that the counter is the only device at this address.
5. On the PNA, press **System** menu, then **Service**, then **Adjustments**, then **10 MHz Freq. Adjust**.
6. Click **Begin Adj.** The application adjusts the internal reference for minimal error and stores the results.
7. Click **Read Freq** to trigger another reading of the 10 MHz signal.

8. Read the current DAC value stored in the analyzer's non-volatile memory (value = 0 - 4095).
9. When the status area indicates the adjustment is complete, click **Exit**.

Procedure for Non-GPIB Counters

1. Connect the counter input to the rear panel 10 MHz Reference Output.
2. Set the counter to at least 1 Hz resolution.
3. If applicable, connect the house-frequency standard to the counter reference input.
4. In the analyzer **System** menu, point to **Service, Adjustments** and click **10 MHz Freq. Adjust**.
5. Under **Frequency Counter**, select **Manual**.



- Adjust the slider bar **arrows** until the frequency counter reads 10.0 MHz at your desired level of accuracy.
- Click **Exit** to save the results.

Data Storage

- The correction data is stored in the EEPROM on the 10MHz Ref board assembly.

Note: If the counter is misreading the frequency, it may be necessary to attenuate the input, or set the input impedance to 50 ohms, or both.

Last Modified:

3-Jan-2014	Updated per JV
20-May-2013	Updated per JV
9-May-2008	Updated with new UI

Display Test

The PNA screen should be bright with all annotations and text readable. The display test allows you to check for non-functioning pixels and other problems.

Note If the display is dim or dark, refer to “Troubleshooting LCD Display Problems” in the PNA Service Guide.

What Is a Damaged Pixel?

A pixel is a picture element that combines to create the image on the display. They are about the size of a small pin point. Damaged pixels can be either “stuck on” or “dark.”

- Stuck on pixel - red, green, or blue; always displayed regardless of the display setting. It will be visible on a dark background.
- Dark pixel - always dark; displayed against a background of its own color.

How to Run the Display Test

On the **System** menu, point to **Service**, and then click **Display Test**.

A multi-color screen is displayed. Be prepared to look for the symptoms described below. Click the Start Test button. To continue to the next test, click the moving Next Test button. The button moves to allow you to see all of the display. After the test is completed, the display defaults to the network analyzer screen.

How to Identify a Faulty Display

One or more of the following indicate a bad display:

- Complete row or column of “stuck on” or “dark” pixels
- More than six “stuck on” pixels (but not more than three green)
- More than twelve “dark” pixels (but not more than seven of the same color)
- Two or more consecutive “stuck on” pixels or three or more consecutive “dark” pixels (but no more than one set of two consecutive dark pixels)
- “Stuck on or “dark” pixels less than 6.5 mm apart (excluding consecutive pixels)

If any of these symptoms occur, your display is considered faulty. See the Service Guide for your PNA model.

Operator's Check

- [Overview](#)
- [How to Run the Operator's Check](#)
- [Operators Check Dialog Box Help](#)

Tip: Use [Move App to Back](#) to cause the PNA application to move behind this application on the screen.

Overview

The Operator's Check should be performed when you first receive your PNA, and any time you wish to have confidence that the PNA is working properly.

Notes

- The Operator's Check does not verify performance to specifications. To verify PNA performance to specifications, run [System Verification](#).
- Allow the PNA to warm up for 90 minutes before considering a failed test to be valid.
- The Operator's Check can NOT be run with a Multiport test set enabled. However, you can run a performance check as described in the Test Set User's Guide. [See the N44xx User's Guide](#).

The Pass/Fail criteria used in the Operator's Check identifies **obvious failures** in the following portions of the PNA hardware:

- Repeatability of the RF switch in the test set
- Attenuation ranges of the test port attenuators (if installed)
- Calibration of the receivers
- Frequency response of the receivers
- Phase lock and leveling
- Noise floor and trace noise

How to Run the Operator's Check

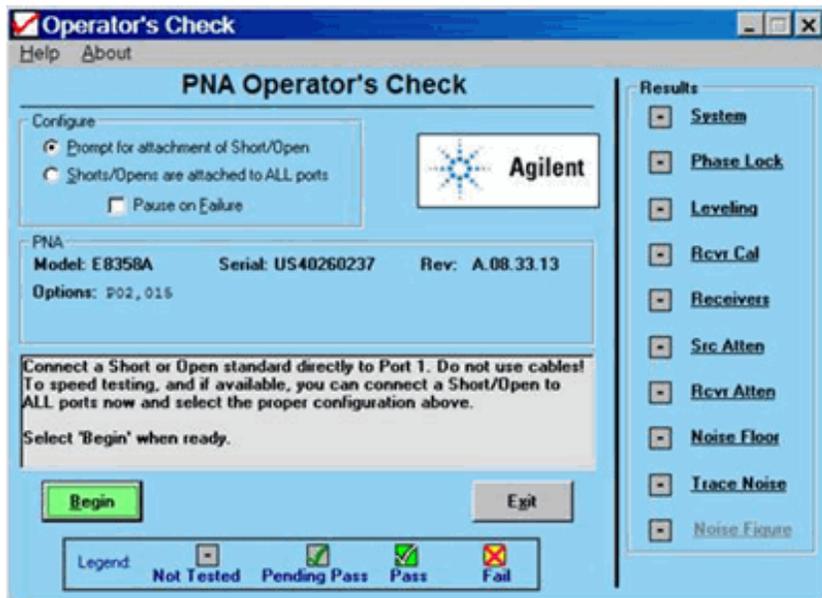
Using front-panel HARDKEY [softkey] buttons

1. Press **SYSTEM**
2. then **[Service]**
3. then **[Operator's Check]**

Using a mouse with PNA Menus

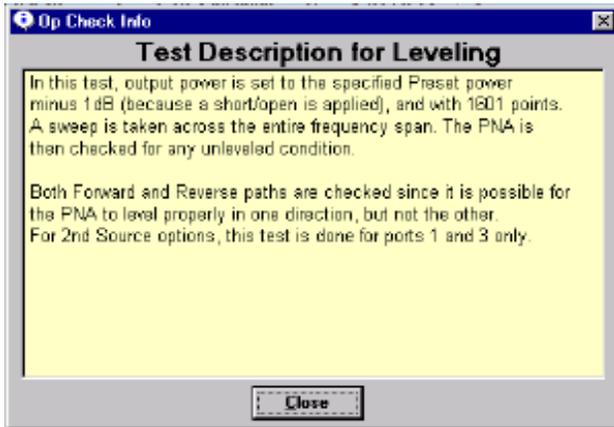
1. Click **Utility**
2. then **System**
3. then **Service**
4. then **Operator's Check**

1. Connect one or more standards (see [Configure](#)).
2. Click **Begin** and **Continue** (if necessary) until "Operator's Check is complete!" appears.



This dialog box will look slightly different, depending on PNA model number and installed options. Some of the tests are not run if the appropriate option is not installed.

To learn about how each test is performed, click one of the tests on the right of the dialog. For example, the following information dialog is launched when **Leveling** is clicked:



Operators Check dialog box help

Note: It is normal for a momentary unlevelled condition to appear during portions of the Operators Check.

Configure

Prompt for attachment of Short / Open If you do not have enough shorts or opens for all test ports, you will be prompted to move the standard to the next test port. Connect either a short or open to port 1, then click Begin.

Shorts / Opens are attached to all ports Connect either a short or open for each test port, then click Begin. All ports are tested without interruption. You can mix shorts and opens on the test ports.

PNA Shows information about the PNA that is being tested.

Legend Shows the status icons used in the Operator's Check and their meaning. **Pending Pass** means that a portion of the testing has been completed successfully.

Results Shows the current status of each test. Click on the test name to learn how that test is performed. This may help in troubleshooting failed tests. If any tests Fail, refer to Chapter 3 of the PNA service guide.

Begin Starts the Operator's Check.

View Results Shows all results in text format. Failed items are preceded by ===>>>.

This text file can be printed or saved with a unique file name to compare results with previous or subsequent testing.

Exit Ends the program and closes the window.

Last Modified:

3-May-2012	Removed C models
4-Sep-2008	Removed legacy content
17-Apr-2008	Added test description note
5-Feb-2008	Added multiport test set note

IF Gain Adjustment

This adjustment sets the IF attenuator/gain stages to present a relatively consistent level to the ADC versus frequency for a given amount of signal. This maximizes the dynamic range of the instrument by preventing any IF compression at higher levels. The adjustment range is about 18dB with one adjustment value per band per input: R1, R2, R3, R4, A, B, C, and D (if applicable.)

The program first finds the peak of each band's frequency response with a fixed input level. It then re-measures those peaks with a power meter. From this information, it determines the optimum gain or attenuation needed to keep all bands as similar as possible for any given frequency.

When to perform

IF Gain Adjustment should be performed when any of the following occur:

- An assembly in the reference receiver path (R1,R2) is replaced.
- The Test Set Motherboard is replaced
- The IF Mux assembly or SPAM assembly is replaced

Required Equipment

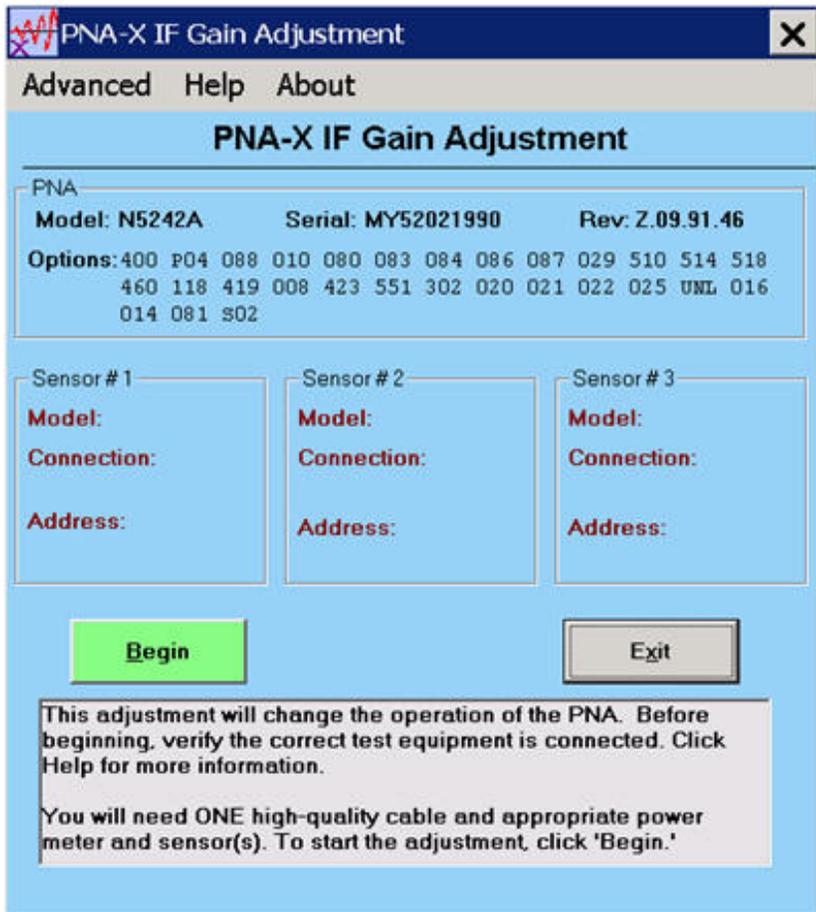
- Power Meter and Sensor(s) - These must cover the full frequency range of the PNA.
- GPIB cable, LAN, or USB cable depending upon the type of Power Meter/Sensor being used.
- Adapters as needed

Procedure

Note: You must be logged onto the PNA as an Administrator to perform an adjustment. [Learn more.](#)

Click **Utilities**, then **System**, then **Service**, then **Adjustments**

At the [Adjustments selection](#), click **IF Gain Adjustment**.



Note: Connect the Power Meter/Sensor to the PNA USB, GPIB, or LAN input depending on the type of Power Meter/Sensor being used. The program will prompt you where and when to attach the RF input for the Power Sensor(s).

Ensure the Reference Channel paths are properly configured and the connections are properly torqued.

1. Click **Begin**
2. Follow the instructions displayed in the program.

The adjustment takes about 5 minutes to complete for a 26.5 GHz PNA. Higher frequency units may take longer.

The Default menu selection (under the advanced menu) is for factory personnel only. This will preset all values to default levels for trouble shooting purposes only. If this is selected, a full IF gain adjustment will need to be performed.

Data Storage

- The correction data is stored in the flash memory on the Test Set Mother Board.

Last modified:

10-Dec-2013	Connection edits per JV
20-May-2013	Updated per JV
3/7/07	Added N5242x information.

System Verification

The performance of the network analyzer is specified in two ways: system specifications, and instrument specifications. It is the end user's responsibility to determine which set of specifications is applicable to their use of the PNA.

A network analyzer measurement "system" includes the analyzer, calibration kit, test cables, and any necessary adapters. The system verification software in the PNA is used to verify the system's conformance to the "system" specifications. A "pass" result demonstrates that the analyzer, test cables, and adapters, perform correctly as a system. It DOES NOT demonstrate that any one component performs according to its individual specifications. A change to any part of this measurement system requires a re-verification of the system.

Instrument specifications specify the network analyzer's uncorrected measurement port characteristics and its output and input behavior. The PNA performance tests are used to verify the analyzer's conformance to "instrument" specifications.

The system verification utility verifies the PNA system specifications by automatically measuring the magnitude and phase for all four S-parameters for each verification device, and comparing the values against the following:

- Factory measured data from files on the verification disk
- Limit lines based on the measurement uncertainty

System Verification requires the use of a calibration kit and verification kit which has been certified within the past 12 months by Agilent. System Verification can NOT be used to perform this kit certification.

[Operator's Check](#) should also be performed to verify the basic operation of the PNA.

- [Equipment Used in the System Verification](#)
- [Precautions for Handling Airlines](#)
- [Flow Diagram of Procedure](#)
- [Procedure for System Verification](#)
- [If the System Fails the Verification Test](#)
- [Interpreting the Verification Results](#)

Notes

- Although the performance for all S-parameters is measured, the S-parameter phase uncertainties are less important for verifying system performance. Therefore, the limit lines will not appear on the printouts.
- System Verification can NOT be run with a Multiport test set enabled. However, you can run a performance check as described in the Test Set User's Guide. [See the N44xx User's Guide.](#)

Equipment Used in the System Verification

N5230A (20 GHz)

(3.5 mm test ports)

N5241A and N5242A

Equipment Type	3.5 mm	Type-N
Calibration kit	85052B/C/D	85054B/D
or		
ECAL Module	N4691A	N4690A
Verification kit	85053B	85055A
RF Cable(s)	Single: 85131C/E Pair: 85131D/F	Single: 85132C/E Pair: 85132D/F
Adapters	None	Single: 85130C and one 7mm-to-Type-N from 85054B cal kit <u>Pair</u> : Two 7mm-to-Type-N from 85054B cal kit

N5230A (40 or 50 GHz)

(2.4 mm test ports)

Equipment Type	2.4 mm	3.5 mm	Type-N
Calibration kit	85056A/D	85052B/C/D	85054B/D
or			
ECAL Module	N4693A	N4691A	N4690A
Verification kit	85057B	85053B	85055A
RF Cable(s)	Single: 85133C/E Pair: 85133D/F	Single: 85134C/E Pair: 85134D/F	Single: 85135C/E Pair: 85135D/F
Adapters	None	<u>Single</u> : 85130F <u>Pair</u> : None	<u>Single</u> : 85130E and two 7mm-to-Type-N from 85054B cal kit <u>Pair</u> : Two 7mm-to-Type-N from 85054B cal kit

E8361A/C
(1.85 mm test ports)

Equipment Type	1.85 mm	2.4 mm 3.5 mm Type-N
Calibration kit	85058B	See 2.4 mm test port table above
or		
ECAL Module	N4694A	
Verification kit	85058V	See 2.4 mm test port table above
RF Cable(s)	Single: N4697E Pair: N4697F	See 2.4 mm test port table above
Adapters	None	See 2.4 mm test port table above

Cable Substitution

The test port cables specified for the PNA have been characterized for connector repeatability, magnitude and phase stability with flexing, return loss, insertion loss, and aging rate. Since test port cable performance is a significant contributor to the system performance, cables of lower performance will increase the uncertainty of your measurement. It is highly recommended that the test port cables be regularly tested.

If the system verification is performed with a non-Agilent cable, ensure that the cable meets or exceeds the operation of the specified cable. Refer to the cable User's Guide for specifications.

Cable Flex Factor

Flex Factor determines how much of the cable phase uncertainty to include in determining the limit

lines.

- Set to **0% (zero)** if the cables are held down in a fixture and are not allowed to move during the calibration and verification.
- Set to **100%** if the cables are allowed to move a lot.

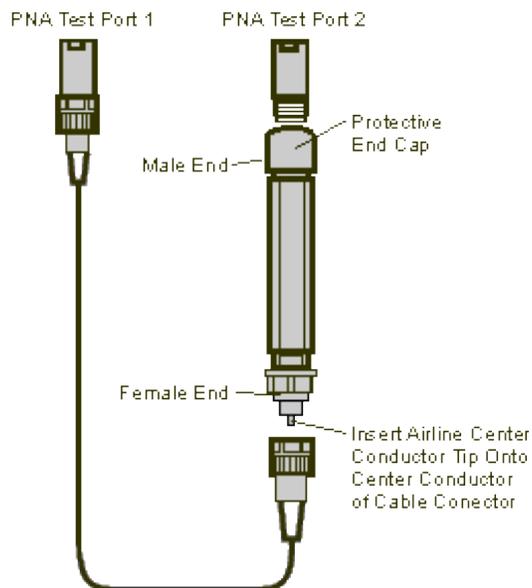
Calibration Kit Substitution

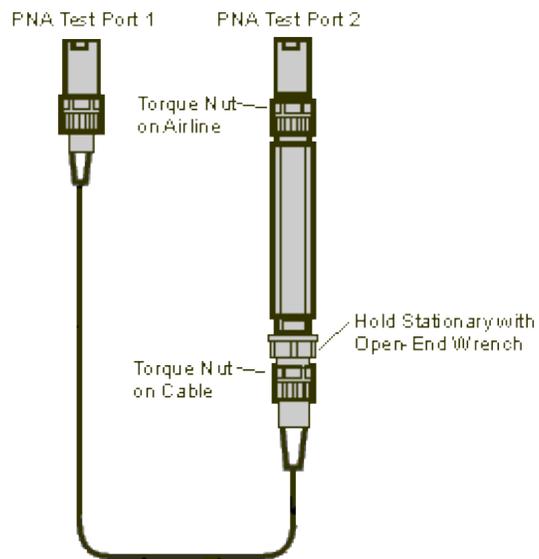
Non-Agilent calibration kits are not recommended nor supported.

Precautions for Handling Airlines

When you are using the airlines in the verification kit, observe the following practices to ensure good measurement techniques.

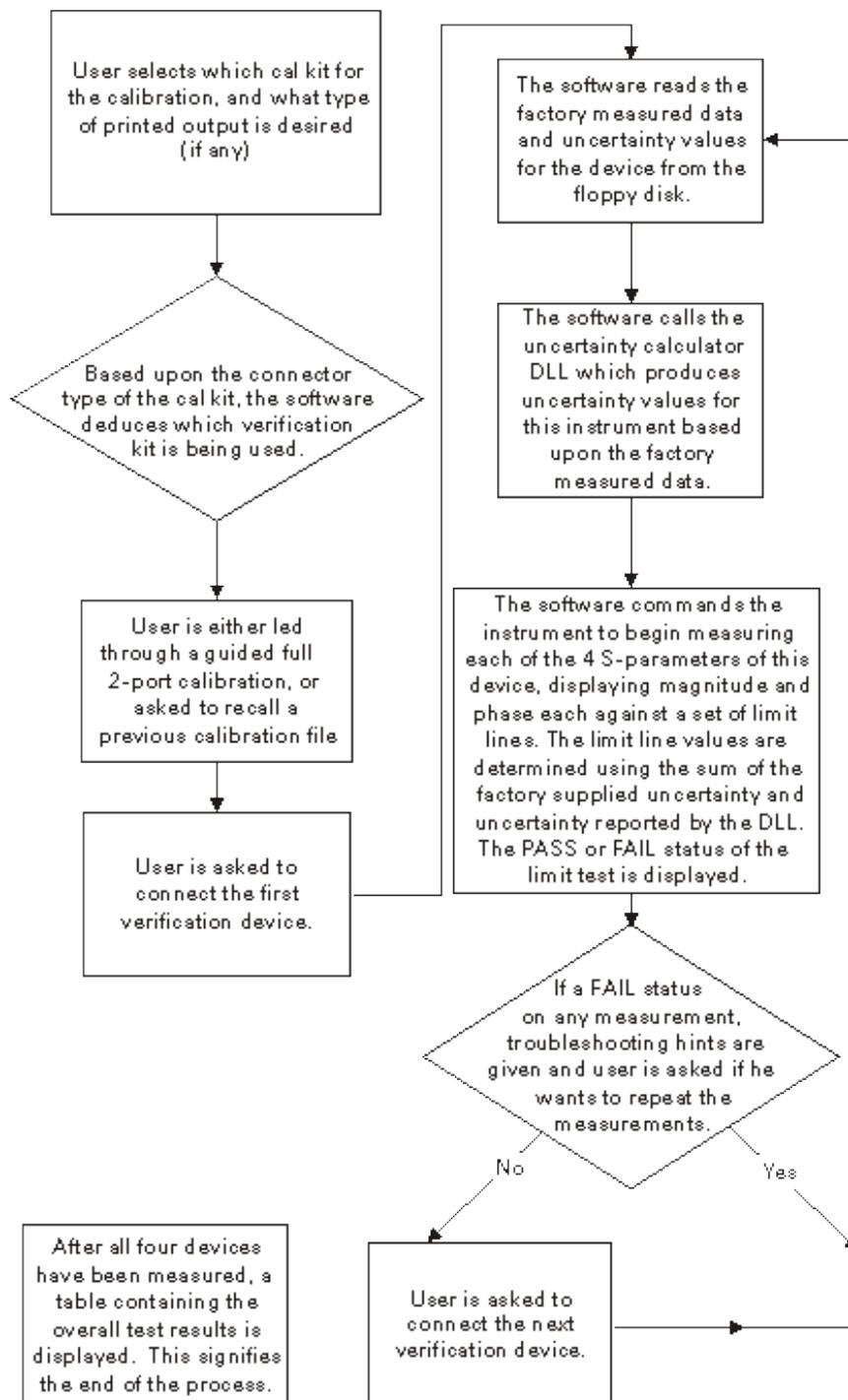
- Be very careful not to drop the airline's center or outer conductor. Damage will result if these devices are dropped.
- Use proper Electro-Static Discharge (ESD) procedures.
- Clean your hands or wear gloves as skin oils will cause a change in electrical performance.





Flow Diagram of Procedure

The operational flow of the software is depicted by the flowchart shown below.

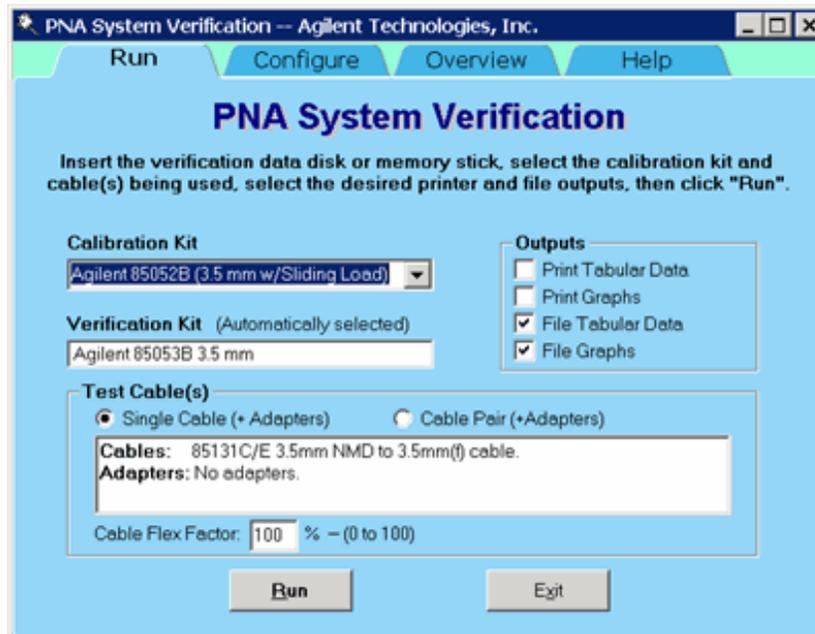


Procedure for System Verification

1. If you want printed test outputs, connect a printer to the analyzer. Let the analyzer warm up for at least 30 minutes.

2. Insert the PNA verification kit floppy disk into the analyzer disk drive.
3. On the **System** (or **Utility**) menu, point to **Service**, and click **System Verification**. The System Verification window similar to this will be displayed.

System Verification Dialog



4. In the **Calibration Kit** box, select the calibration kit or ECal module that is being used. The corresponding verification kit to use appears in the **Verification Kit** box.
5. Under **Printer Output** click on any of the following options.
 - o **Print Tabular Data:** Prints the verification data in tabular form which includes measured data and uncertainty limits. Refer to a tabular data example, later in this topic.
 - o **Print Graphs:** Prints the verification data in graphical form. The graphic form includes the measured data trace, factory supplied data trace and uncertainty limits. Refer to a plot data example, later in this topic.
 - o **File Tabular Data:** Writes the verification data in tabular form to a text file in the c:\users\public\network analyzer\documents directory.
 - o **File Graphs:** Saves a screen image in .PNG format in the c:\users\public\network analyzer\documents directory.

Note: If you want printed output, it is assumed you have already installed the Windows driver for your particular printer, and have tested that you can print to the printer from the network analyzer. This software is designed to print to whichever printer is currently set as the Default printer (see Printers in the Windows Control Panel).

6. To modify the number of ports to be verified, to change the number of devices to measure, or to use a previously stored verification calibration, click on the **Configure** tab and make the desired selections.
 - o For the system verification to be truly adequate, the software must measure all devices in the kit with a recent calibration applied. Removing and reattaching any test port cables or adapters invalidates all previous calibrations.
7. Click **Run**.
8. Follow the instructions on the analyzer for performing the system verification, inserting the verification devices as prompted.

Note for 3 Port PNA:

The System Verification Procedure is **repeated three times**. The first time, **Ports 1 and 2** are measured as a pair; then **Ports 1 and 3** are measured; and lastly, **Ports 2 and 3** are measured.

Note for 4 Port PNA:

The System Verification Procedure is **repeated two times**. The first time, **Ports 1 and 2** are measured as a pair, then **Ports 3 and 4** are measured.

Step-by-Step Process Description

1. Depending upon the selected choice in the Calibration submenu of the Configure menu, the user is either prompted to recall a previous calibrated instrument state, or is guided through a full 2-port calibration using the selected calibration kit. For ECal, the ECal module is connected just once; a standby message is posted while the software is performing the calibration.
2. The user is prompted to connect the first verification device.
3. The software reads the factory measured data for that device and uncertainty values for that data (CITfiles) from the floppy disk supplied with the verification kit.
4. The software sends the factory measured data, calibration kit and instrument state information to the uncertainty calculator DLL, which generates uncertainty values specific to the PNA.
5. The analyzer first sets up for magnitude measurements of all four S-parameters, each parameter in a separate window (lin mag for S_{11} and S_{22} , log mag for S_{21} and S_{12}). Each of the factory measured S-parameters are fed to the appropriate window as a memory trace. Limit line offsets are calculated as the sum of the factory measured data uncertainties and PNA uncertainties reported by the DLL. Upper and lower limits are displayed (factory measured data + uncertainty sum, factory measured data - uncertainty sum). The PNA takes a sweep, limit test is turned on and PASS/FAIL status is reported in each of the four windows.
6. The user clicks a button when ready to view phase measurements. The four windows get updated for phase format, phase memory traces, phase limits and PASS/FAIL result.
7. If the limit test of any of the four S-parameters (magnitude or phase) indicates a FAIL status, the software suggests troubleshooting tips and asks if the user would like to repeat measurement of that device or proceed to the next device. If proceeding to the next device, the factory measured data and uncertainties for the next device are read from floppy, the uncertainty DLL gets called with this next set of factory measured data, and the four measurement windows get updated for magnitude measurement of the next device.

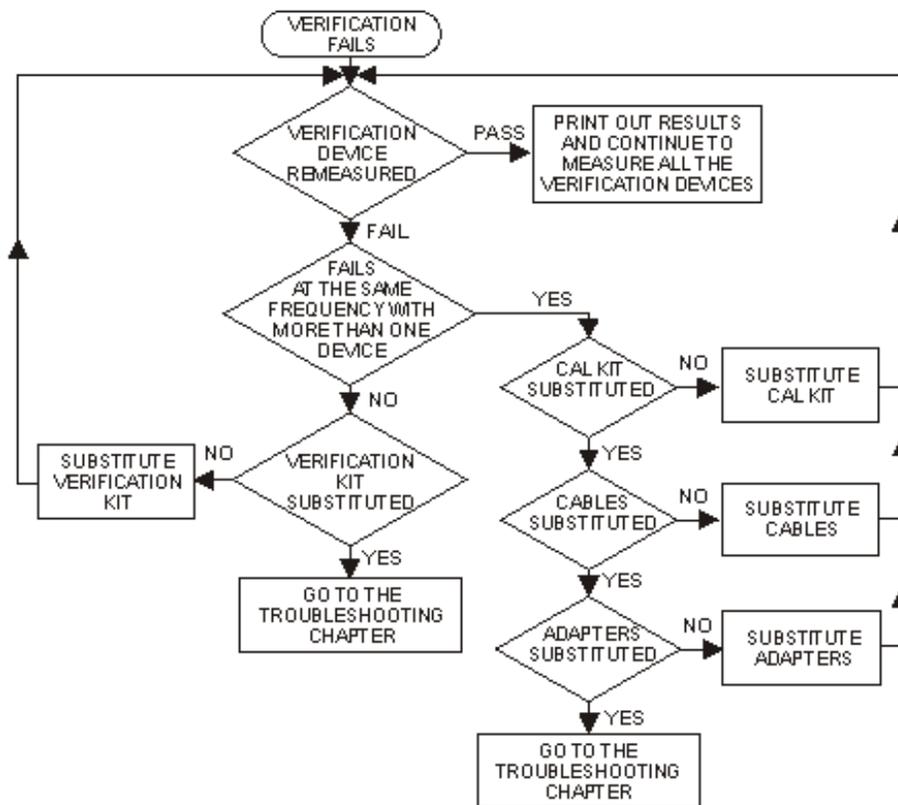
- The software follows this same process until all selected devices have been measured, at which point a summary window is displayed containing the set of PASS/FAIL results for all four parameters of each device.

If the System Fails the Verification Test

IMPORTANT: Inspect all connections. Do not remove the cable from the analyzer test port. This will invalidate the calibration that you have done earlier.

- Repeat this verification test. Make good connections with correct torque specifications for each verification device.
- Disconnect, clean and reconnect the device that failed the verification test. Then measure the device again.
- If the analyzer still fails the test, check the measurement calibration by viewing the error terms as described in "Front Panel Access to Error Terms" on page 4-7 of the Service Guide.
- Refer to the graphic below, for additional troubleshooting steps.

Verification Fails Flowchart



Interpreting the Verification Results

The graphic below shows an example of typical verification results with **Tabular Data** selected in the **Printer Output** area of the **System Verification** window. A graphic later in this topic shows an example of typical verification results with **Measurement Plots** selected in the **Printer Output** area of the **System Verification** windows. These printouts include a comparison of the data from your measurement results with the traceable data and corresponding uncertainty specifications. Use these printouts to determine whether your measured data falls within the total uncertainty limits at all frequencies.

The tabular data consists of:

- Frequency of the data points (in MHz).
- Lower limit line as defined by the total system uncertainty specification.
- Results of the measurement.
- Upper limit line as defined by the total system uncertainty specification.
- Test status (PASS or FAIL) of that measurement point.

Printout of Tabular Verification Results

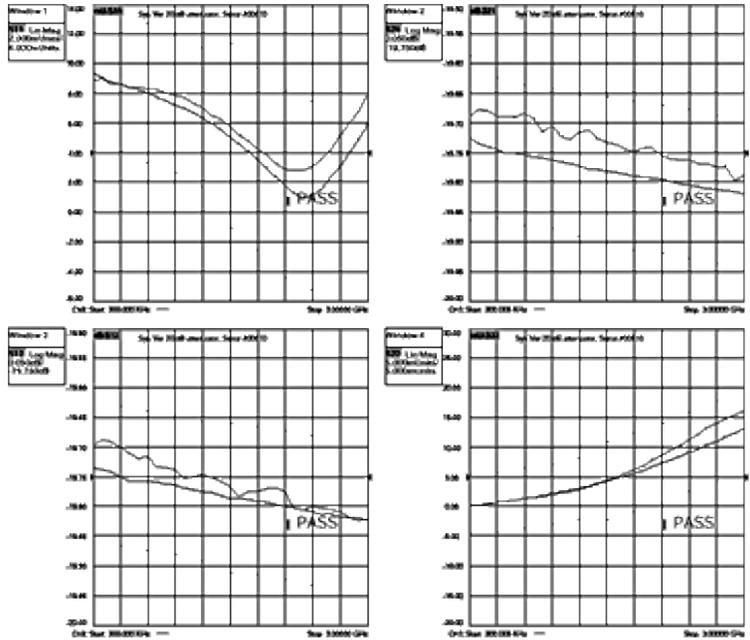
The image shows two overlapping printout pages from a system verification window. The pages display tabular data for verification results. The tables are organized into columns, including frequency (MHz), measured values, and test status (PASS or FAIL). The data points are listed for various frequencies, and the test status is indicated for each measurement point. The printouts include a comparison of the data from your measurement results with the traceable data and corresponding uncertainty specifications.

The printed graphical results show:

- Upper limit points as defined by the total system uncertainty specifications.
- Lower limit points as defined by the total system uncertainty specifications.

- Data measured at the factory.
- Results of measurements.
- Measurement parameter names and formats (Lin Mag or Log Mag).
- Serial number of device (00810).
- Device being measured (Sys Ver 20 dB attenuator).

Printout of Graphical Verification Results



Last Modified:

- 6-Apr-2009 Updated for N5241A
- 8-Apr-2008 Updated for 'C' Models
- 11-Feb-2008 Added note about multiport
- 15-Jan-2008 Added Flex Factor and image

Source Adjustment

Source Adjustment is a **SERVICE** Routine which should be performed when a component in the source chain is replaced, or when the PNA fails an annual calibration. It adjusts the PNA source power for flatness across its full frequency range.

This topic does **NOT** discuss [Source Power Calibration](#), which calibrates a PNA source over the current measurement range.

Required Equipment

Note: The power sensor depends on the PNA frequency range. Depending on the PNA model, two power sensors may be required to test the full frequency range. The PNA front panel connector type will determine the cable used and if an adapter is required with the power sensor(s).

[See list of supported power meters and sensors.](#)

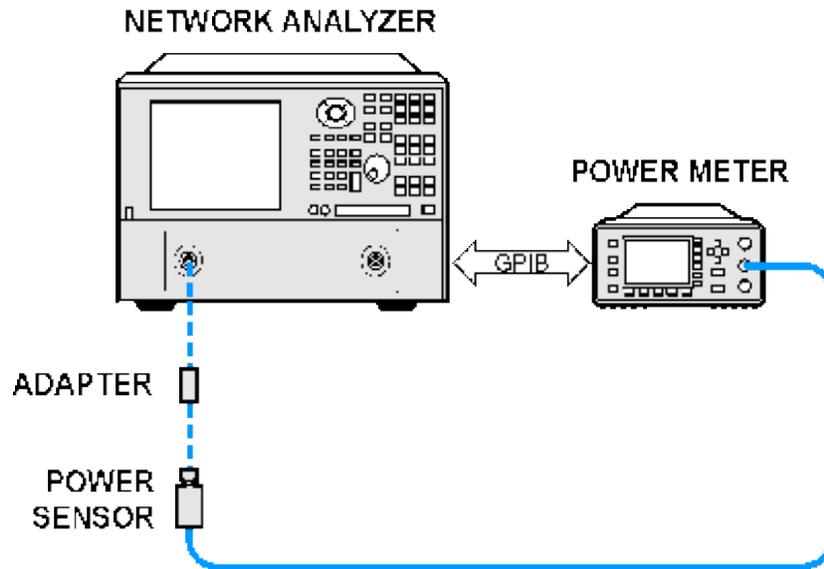
[See PNA Accessories](#)

Procedure

Note: You must be logged onto the PNA as an Administrator to perform an adjustment. [Learn more.](#)

1. Refer to your power meter documentation to ensure the proper calibration factors for the power sensor have been entered into the table on the power meter.
2. Connect a GPIB cable between the power meter and network analyzer (use the System Controller GPIB port if applicable.)
3. Ensure the power sensor(s) are connected to the power meter.
4. Click **Utilities**, then **System**, then **Service**, then **Adjustments**
5. At the [Adjustments selection](#), click **Source Adjustment**
6. There are 3 different version of the Source Calibration software; all are slightly different. All have a button that is labeled "Calibrate" or "Adjust". This is the button that will begin the calibration process. Some versions will also have a button labeled "Verify" that will test the source calibration without making any changes. Other selections are for factory personnel use only.
7. Once begun, you must enter the power meter and sensor information. The software will verify the power meter and sensor. You are then prompted to connect the sensor(s) and cable as needed.

Connecting sensors to the PNA



Additional Information

All ports are tested on all PNAs. Source calibration takes approximately 10 to 45 minutes to complete depending on the frequency range and model number of the PNA.

Troubleshooting

In the event there is a problem with Source Adjustment, please refer to the "Troubleshooting" chapter in the PNA Service Guide.

Data Storage

- The correction data is stored in the flash memory on the Test Set Mother Board.

Last Modified:

22-May-2013	Updated info from JV
6-Apr-2009	Updated models
10-Mar-2009	Changed to Source Adjustment

Receiver Adjustment

This program adjusts the network analyzer receivers for a flat response across its full frequency range. This adjustment is for service only; not for measurement calibration.

Required Equipment

[See list of supported power meters and sensors.](#)

See [PNA Accessories](#)

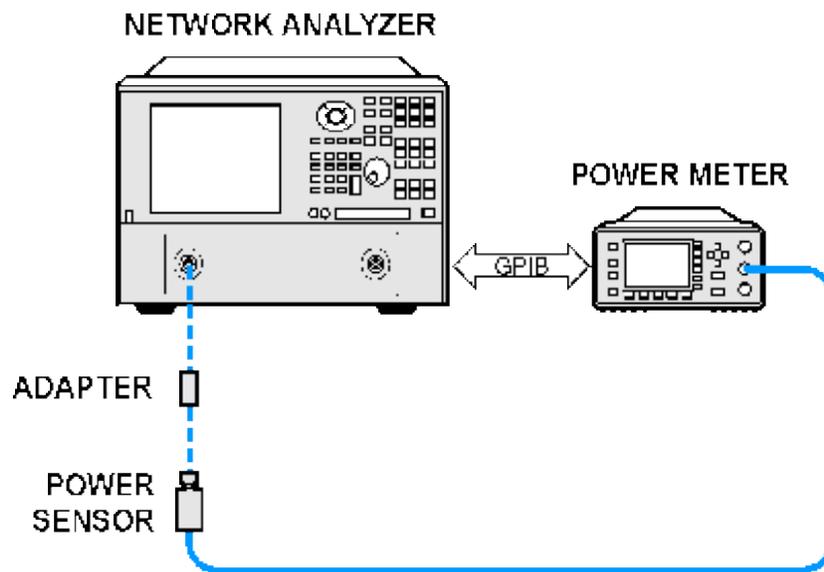
Notes

- The power sensor depends on the PNA frequency range. Depending on the PNA model, two power sensors may be required to test the full frequency range. The PNA front panel connector type will determine the cable used and if an adapter is required with the power sensor(s).
- In this adjustment, a power sensor with a specified lower frequency limit of 50 MHz may be used on all instrument that have a lower frequency limit of 10 MHz. Any added uncertainty is negligible.
- If using an older style sensor (without built-in correction factors), refer to your power meter documentation to ensure the proper calibration factors for the power sensor have been entered into a table on the power meter.
- You must be logged onto the PNA as an Administrator to perform an adjustment. [Learn more.](#)

Procedure

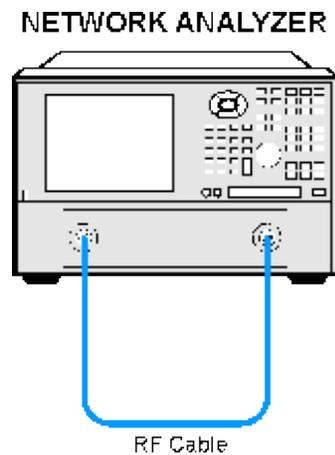
1. Click **Utilities**, then **System**, then **Service**, then **Adjustments**
2. At the Adjustments selection, click **Receiver Adjustment**.
3. Connect a GPIB cable between the power meter and network analyzer.
4. Ensure the power sensor(s) are connected to the power meter.
5. The software presents you with two choices:
 - a. Click **Inspect Flatness** to observe flatness of receiver response versus frequency. Although there is no explicit specification for receiver flatness, Receiver Calibration should improve Transmission and Reflection Tracking error terms which are specified.
 - b. Click **Calibrate** to begin the receiver calibration process. The software prompts you to connect the sensor(s), cable and adapter as needed (see the following graphics).

Connecting sensor(s) to the PNA



Connecting adapter and cable between sensor and PNA

Through connection using the specified cable



Additional Information

- Receiver Adjustment tests all PNA receivers, taking approximately 15 and 45 minutes. Length is dependent on frequency range and number of ports.
- Upon completion of the Receiver Adjustment, a transmission measurement of a good quality cable should appear to be smooth, with slightly increasing loss versus frequency. A reflection measurement of a short or open should appear to be a flat line across the entire frequency range with only a dB or two of variation/ripple. If instead, you see variations of 10-40dB, then the PNA may have a mixer problem. Typically, this means the uncorrected low end frequency phase relative to other receivers is different. This causes the correction algorithm to "blow up" and provide wildly incorrect data. This is almost always a hardware problem and typically one or more receivers must be replaced. The adjustment procedure has a quick test for this and it will show a warning message if excess phase shift is detected, however this test is not definitive and may not always catch every problem.

Troubleshooting

In the event there is a problem with Receiver Adjustment, please refer to the "Troubleshooting" chapter in your PNA Service Guide.

Data Storage

- The correction data is stored in the flash memory on the Test Set Mother Board.
-

Last Modified:

21-May-2013	Changed cal to adjustment.
12-Jul-2007	Removed outdated table of supported power meters

Receiver Display

- [The Receiver Display as a Troubleshooting Tool](#)
- [How to start the Receiver Display](#)

Other Support Topics

The Receiver Display as a Troubleshooting Tool

The Receiver Display is a Troubleshooting Tool. It enables the analyzer to isolate faulty functional groups within its own Measurement System. Traces for each Receiver are Displayed in individual windows. Identifying discrepancies of the traces in these windows can help isolate the faulty assembly.

For a thorough description of Receiver Display and the troubleshooting steps see Chapter 3 of the PNA Service Guide. You can download the Service Guide for your PNA model from our website:

<http://na.tm.agilent.com/pna/>

How to Start the Receiver Display

Using front-panel HARDKEY [softkey] buttons

1. Press **SYSTEM**
2. then **[Service]**
3. then **[Utilities]**
4. then **[Receiver Display]**

Using a mouse with PNA Menus

1. Click **Utilities**
2. then **System**
3. then **Service**
4. then **Utilities**
5. then **Receiver Display**

[Programming Commands](#)

IF Path Configuration Settings

This dialog is used to set many IF receiver settings for models.

In this topic:

- [PNA-X and N522xA IF Frequencies](#)
- [How to Start the IF Path Configuration dialog](#)
- [IF \(Receiver\) Path Configuration dialog box](#)
- [Expanded Block Diagram and Descriptions](#)
- [IF Path Configuration using COM and SCPI](#)

Other IF Access Topics

Auto IF Frequencies

Wideband/Normal IF path:

Note: For the following discussion, **RF** = Receiver Frequency

With DSP Version 4:

- $RF < 53 \text{ MHz: IF} = 2.535211 \text{ MHz} \quad [3 \times (60e6 / 71)]$
- $RF \geq 53 \text{ MHz: IF} = 7.605634 \text{ MHz} \quad [9 \times (60e6 / 71)]$

With DSP Version 5, the IF frequency is dependent on the RF AND the current IFBW setting:

- All RF; IF Bandwidth $\geq 1\text{MHz}$: (All Models)

IFBW Setting	IF Frequency
1 MHz	7.692 MHz
1.5 MHz	7.368 MHz
2 MHz	8.450 MHz
3 MHz	8.163 MHz
5 MHz	6.897 MHz
7 MHz	10.53 MHz
10 MHz	15.38 MHz
15 MHz	22.22 MHz

- IF Bandwidth <= 600 kHz:
 - RF >= 53 MHz; All models: IF = 7.438017 MHz $[(9 \times (100e6 / 121))]$
 - RF <= 53 MHz; PNA-X models: IF = 2.479339 MHz $[(3 \times (100e6 / 121))]$
 - RF <= 53 MHz; N522xA models: IF = 826.446 kHz $[1 \times (100e6 / 121)]$

Narrowband IF path:

- IF = 10.70 MHz
- Bandwidth = 30 kHz

Manually change the IF frequency

The IF frequency can be changed to any value between +14.9999 MHz and -14.9999 MHz using [SENS:IF:FREQ](#) or [IFFrequency](#) commands.

- With DSP Version 4 - 34 and above, min and max IF frequencies up to +/- 20.1 MHz are available.
- With DSP Version 5, min and max IF frequencies up to +/- 38 MHz are available.
- Performance is degraded drastically above +/- 14.9999 MHz.

[Learn about DSP Version](#)

How to start the IF Path Configuration dialog

To provide quicker access, use the Setup softkey. [Learn how.](#)

Using front-panel HARDKEY [softkey] buttons

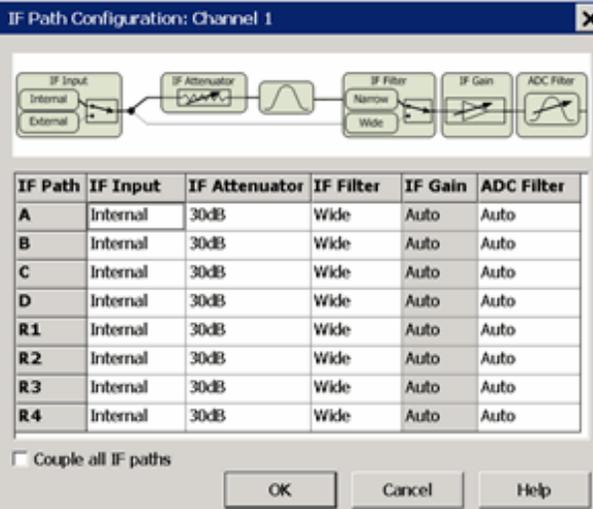
1. Press **CHANNEL**
2. then [**Hardware Setup**]
3. then [**IF Config**]

Using a mouse with PNA Menus

1. Click **Trace/Chan**
2. then **Channel**
3. then **Hardware Setup**
4. then **IF Config**

Programming Commands

IF (Receiver) Path Configuration dialog box help



The screenshot shows the 'IF Path Configuration: Channel 1' dialog box. At the top, there is a block diagram of the IF path with components: IF Input (Internal/External), IF Attenuator, IF Filter (Narrow/Wide), IF Gain, and ADC Filter. Below the diagram is a table with columns: IF Path, IF Input, IF Attenuator, IF Filter, IF Gain, and ADC Filter. The table lists configurations for receivers A, B, C, D, R1, R2, R3, and R4. At the bottom, there is a checkbox for 'Couple all IF paths' and buttons for 'OK', 'Cancel', and 'Help'.

IF Path	IF Input	IF Attenuator	IF Filter	IF Gain	ADC Filter
A	Internal	30dB	Wide	Auto	Auto
B	Internal	30dB	Wide	Auto	Auto
C	Internal	30dB	Wide	Auto	Auto
D	Internal	30dB	Wide	Auto	Auto
R1	Internal	30dB	Wide	Auto	Auto
R2	Internal	30dB	Wide	Auto	Auto
R3	Internal	30dB	Wide	Auto	Auto
R4	Internal	30dB	Wide	Auto	Auto

The IF path, represented in the block diagram at the top of the dialog, is duplicated for each of the receivers (A, B, C, D, R1, R2, R3, R4). In addition, each path can be configured differently for each channel.

Element - Indicates an element in the [expanded block diagram](#).

IF Input (1) - Available on the [PNA-X and N522xA with Opt 020](#). Internal input is a test port or reference receiver input. External Input is through the [PNA-X and N522xA rear-panel connectors](#).

IF Attenuator (3) - Specify IF attenuation for the narrowband path of the selected receiver.

IF Filter (2) - Select Wideband or Narrowband (includes the ability to pulse gates).

IF Gain (10)- Set to **Auto** by default, the following are reasons to change the IF Gain:

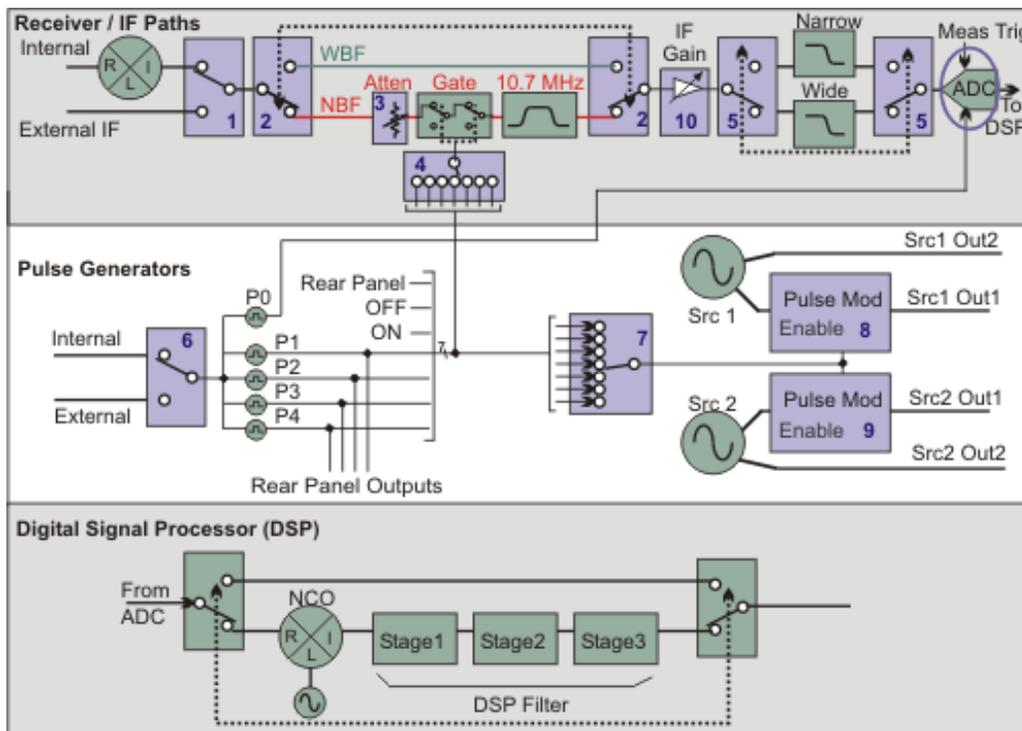
- For [millimeter systems](#) that do NOT use the external millimeter test set, the millimeter head output IFs are routed directly to the RF receivers. You may want to change the IF gain to improve the noise figure of the receivers.
- External couplers are often used for high-power test setups. The PNA automatically adjusts the IF Gain depending on the frequency of operation in order to correct for several aspects of the hardware, including the large coupled-arm roll off at low frequencies (below 700MHz). If you replace the internal coupler with one having a different low-frequency roll off, then you may also want to change the IF gain to avoid overdriving the receiver.
- When using the rear panel direct IF inputs, the gain is set low when in standard operation and very high when in [millimeter mode](#). You may want to control the gain of the direct IF inputs to improve noise figure or to avoid overdriving the receivers.

ADC Filter (5) - Select **Auto**, **Narrow** (9 MHz or 11 MHz), or **Wide** (16 MHz or 38 MHz) filter.

[Learn more.](#)

Couple all IF paths - Check to make the same setting for all receivers.

Expanded Block Diagram (IF / Pulse Generators / DSP)



Blue boxes are configurable elements.

Click a blue box, or scroll down, to see how to make settings using [SCPI](#) and [COM](#) commands.

Receiver / IF Path (top block)

[Scroll up](#) for descriptions of the Receiver / IF Paths blocks. Most of these elements can be set from the front-panel User Interface (UI).

ADC: (Analog to Digital Converter) This block, responsible for quantifying receiver measurements, is triggered when BOTH the Meas Trig line AND P0 pulse generator line are TRUE. In addition, the Meas Trig signal MUST be TRUE **before or at the same time** as the P0 signal.

Meas Trig Signal

- When [PNA Trigger source](#) is set to **Internal**, the Meas Trig line is ALWAYS TRUE. Internal trigger source is the best setting for making Pulse measurements. This means that the P0 line largely determines when and what pulse measurements are made.
- When PNA Trigger source is set to **External**, the Meas Trig line can be configured from the [Meas Trig \(External\) dialog](#).

P0 (Trig) Signal - [See below](#).

Pulse Generators (middle block in [above diagram](#))

[See how to make these settings remotely](#)

Switch 6: Represents **Internal** or **External** triggering for the pulse generator. This setting, and the External Trigger Pulse settings, are made on the Trigger dialog, [Pulse Trigger tab](#).

On the Integrated Pulse App, [Pulse Setup dialog](#), when **Master** Pulse Trigger is set to External or an external pulse generator, then External is selected here automatically.

- **Internal** - The pulse generator is internally triggered and puts out a periodic pulse train with a period defined by the [pulse generator settings](#).
- **External** - The pulse period is ignored, and the pulse generator puts out one set of pulses (P0-P4) per external trigger. All five pulse outputs have unique [delay and pulse width settings](#).

External trigger input is on the [Pulse I/O connector](#) pin 7 (PulseSyncIn). The PulseSyncIn line provides a configurable trigger signal into the Pulse Generators. If a level trigger is still valid when the first pulse set is finished, another set will be generated. Only one set of pulses is emitted when edge triggering is used. The length of time that it takes to emit one set of pulses is the end time of the last enabled pulse (largest of width + delay of all the pulses P0-P4.)

The External pulse input polarity (positive or negative) and type (edge or level) is configurable only with DSP version: 4.0 FPGA: 34 or higher. [Learn more](#). Otherwise, the pulse generators respond only to positive, level input trigger signals.

P0: When P0 is enabled, it is hardwired to trigger the data acquisition ADCs. [See ADC \(above\)](#).

- If the data acquisition system is not ready (Meas Trig = NOT TRIGGERED), the P0 trigger is ignored.
- If the pulse generator is internally triggered, then the data acquisition system receives periodic triggers.
- If the pulse generator is externally triggered, then the data acquisition system receives a trigger each time the pulse generator is triggered.

In either case, data acquisition is synchronized to the pulse generator ONLY when Sync is checked or Pulse0 is enabled (exactly the same thing). Data acquisition begins on the rising edge of P0. The width of P0 does NOT directly matter as data acquisition does not stop when P0 goes false. The width of P0 DOES matter when determining the time for one set of pulses when externally triggered (see [External trigger](#) above). The following describes how the P0 generator triggers data acquisition:

- **Step mode sweeps** of any sweep type: By default, each P0 rising edge triggers a single data point. When [point averaging](#) is on, all of the measurements (subpoints) that are required to average each point are made with a single trigger. To individually trigger the acquisition of each subpoint, send the subPointTrigger ([SCPI](#) or [COM](#)) command.
- **CW sweeps:** Each trigger initiates acquisition for the entire sweep. This is currently used for wideband pulse profiling.

P1 thru P4 These four pulse generator outputs are hardwired to rear panel outputs on the [Pulse I/O connector](#) (pins 10 - 13). They are also routed to two switches (#4 and #7 on the above diagram) along with the following three lines:

- **Rear Panel** External pulse generator input from [Pulse I/O connector](#) pin 8 (RFPulseModIn).
- **OFF** Pulse is constantly in LOW state causing gate and source to be OFF.
- **ON** Pulse is constantly in HIGH state causing gate and source to be ON.

Switch 7 Pulse Modulation - 1 of 7 lines to each of the sources. **Important:** When internally modulating the sources, [source leveling must be set to Open-loop](#).

Rear-panel Outputs: [Pulse I/O connector](#) (pins 10 - 13) hardwired.

Source1 and Source2 pulse modulators: (#8 and #9 on the above diagram)

DSP (bottom block)

- Filters the ADC (digital) output from top block and outputs data to the PNA display.
- See [SCPI](#) and [COM](#) commands to control DSP settings.

See Also

- [Pulse Settings](#)
- [Remote RF Path Configuration](#)
- [Rear Panel Pulse I/O connector](#)

IF Path Configuration using COM and SCPI

Most of the following elements, [highlighted in BLUE in the above Block Diagram](#), have settings that are made using SCPI or COM commands. In general, the command specifies an element name and a setting.

- [See SCPI command](#)
- See [COM object](#) and [example](#).
- These are the same commands that are used to make settings in the [RF Path Configurator](#).

Ref#	Element Name	Settings
	Description	
1	"IFSWn" For 2-port PNA-X and N522x, n = A, B, R1, R2 For 4-port PNA-X and N522x, n = A, B, C, D, R (for R1 to R4) For example: "IFSWB" Requires Opt 020 external IF inputs on the rear panel	"Internal" "External" Rear Panel IF connectors . 4-port use R for Ref 1 to 4

<p>2</p>	<p>"IFPathn" For 2-port PNA-X and N522x, n = A, B, R1, R2 For 4-port PNA-X and N522x, n = A, B, C, D, R1, R2, R3, R4 "IFSigPathAll" makes setting for ALL receivers.</p>	<p>"WBF" Wide Band Filter Path (default) "NBF" Narrow Band Filter Path</p>
<p>3</p>	<p>"NBFATNn" For 2-port PNA-X and N522x, n = A, B, R1, R2 For 4-port PNA-X and N522x, n = A, B, C, D, R1, R2, R3, R4 For example: "NBFATNB"</p>	<p>0 to 31 in 1 dB steps For example: "28"</p>
<p>4</p>	<p>"IFGaten" For 2-port PNA-X and N522x, n = A, B, R1, R2 For 4-port PNA-X and N522x, n = A, B, C, D, R1, R2, R3, R4 For example: "IFGateB"</p>	<p>"On" Gate is always ON "Off" Gate is always OFF "RearPanel" (use Pulse IO pins 1 to 5) "Pulse1" "Pulse2" "Pulse3" "Pulse4"</p>
<p>5</p>	<p>"IFAntiAliasFilter" <ul style="list-style-type: none"> This filter is labeled ADC Filter on the IF Path Configuration dialog. This setting affects ALL receivers. It can NOT be made for individual receivers. The ADC Filter values depend on the DSP Version. Learn more. </p>	<p>"Auto" PNA selects which filter to use based on other IF settings. "Narrow" Sets 9 MHz for DSP Version 4 Sets 11 MHz for DSP Version 5 "Wide" Sets 16 MHz for DSP Version 4 Sets 38 MHz for DSP Version 5 "9MHZ" is Superseded - Use "Narrow".</p>

		"16MHZ" is Superseded - Use "Wide".
6	"PulseTrigInput" Requires Opt 025 - Four Internal Pulse Generators	" Internal " Internal Pulse In - pulse generators are triggered each period. " External " External Pulse Synchron - Pulse I/O pin 7) - An external trigger signal is required to trigger the pulse generators for each pulse. <name> An external pulse generator to be configured as the Master Pulse Trigger.
7	"PulseModDrive" Select from 1 of 7 lines to modulate the OUT1 path of Sources 1 and 2 . Important: When Pulse 1-4 is selected to modulate the sources, source leveling must be set to Open-loop .	" On " Pulse Mod drive is always ON, leaving "SRC1 2 Out 1" ON and not modulated. Default setting. " Off " Pulse Mod drive is always OFF, leaving "SRC1 2 Out 1" OFF. " RearPanel " (use Pulse IO pin 8) " Pulse1 " " Pulse2 " " Pulse3 " " Pulse4 "
8	"Src1Out1PulseModEnable" Requires Opt 021 - Source1 Pulse Modulator	"Enable" "Disable"
9	"Src2Out1PulseModEnable" Requires Opt 022 - Source2 Pulse Modulator	"Enable" "Disable"
10	"IFGAIN""n For 2-port models, n = A, B, R1, R2	"0", "2", "4", "6", "8", "10", "11", "13", "15", "Auto"

For 4-port models, $n = \mathbf{A, B, C, D, R1, R2, R3, R4}$

For example: "IFGAINB"

Last Modified:

11-Jul-2013	Provided ink to block diagram
21-Dec-2011	Fixed IFPathN
7-Sep-2011	More sync info and image from NJ.
10-May-2011	Added N522xA models
25-Aug-2010	Modified for new DSP version and IF Gain (A.09.30)
8-Mar-2010	Modified for UI (A.09.20)
9-Dec-2009	Added external pulse input polarity and type
20-Jul-2009	Added subpointtrig content
5-Sep-2007	New Image and minor edits
5-Feb-2007	MX New topic

External Millimeter-Wave Module Configuration

This feature, when used with the N526xA test sets and external mmWave Modules, extends the frequency coverage of your PNA. The **N5251A** broadband system is also configured using the [Millimeter Module Configuration dialog](#).

In this topic:

- [Features and Limitations](#)
- [How to Configure Millimeter Modules](#)
- [Mixer Mode](#)
- [mmWave Module Power Level Control](#)

See Also

- [N5261A / N5262A User Manual](#) (Requires internet connection).
- [N5251A Installation Guide](#) (Requires internet connection).
- [Millimeter-Wave Network Analyzers Technical Overview](#) (Requires internet connection)
- [mmWave Measurements with No Test Set](#)
- [Download a macro for Configuring VDI Frequency Extenders](#) (Requires internet connection)

Other IF Access Topics

Note: In the PNA user interface and in this help file, the N526xA Millimeter Head Controller is referred to as a **test set**. OML test head modules are referred to as **mmWave modules**.

CAUTION: Turn OFF test set power before connecting or disconnecting the DC cable to the mmWave modules.

Features

- Controls N5260A, N5261A, and N5262A Test Sets.
- Compatible with [iTMSA \(True Mode Stimulus\)](#).
- Several methods available to provide [Leveled power to the DUT Input](#)
- Compatible with [Integrated Pulse Application](#)

The following [Applications](#) are supported:

- [SMC mixer measurements](#).
- [IM Spectrum](#) and [IM Spectrum for Converters](#)

The following configurations are supported:

- PNA-X or N522xA with options [Opt 020](#) works with N5260A, N5261A, and N5262A Test Sets.
- When using the N5262A test set, a 2-port PNA-X or N522xA requires [Opt 551](#).

Limitations

- **Power Settings** When using mmWave modules with an N5260A test set, the PNA can NOT control the power level into your DUT. Your mmWave modules may have a variable attenuator on them. When used with an N5261A or N5262A, after performing a Source Power Cal, then the PNA power settings may be used to control the power into the DUT. See [Leveled Power Capabilities](#).
- To protect your mmWave modules from damage, the settings on the Millimeter Module Configuration dialog can ONLY be changed manually. They can NOT be reset or changed by performing a Preset, by recalling an Instrument State, or from a remote program.
- ONLY the [Applications](#) listed about are supported.

PNA-X Notes

- **CAUTION:** Connect a 10 dB attenuator to the N5260A LO input from the [LO Output](#). Otherwise, damage will occur to the N5260A test set.
- The PNA-X or N522xA rear panel [IF Inputs use 5 SMA connectors](#). Previous PNA models use BNC connectors. Adapters may be required.
- Beginning with A.09.00, Frequency Offset and SMC Measurements are supported when using mmWave modules. [Learn more](#).

How to Configure Millimeter-Wave Modules

Using front-panel HARDKEY [softkey] buttons

1. Press **SYSTEM**
2. then [**Configure**]
3. then [**Millimeter Module**]

Using a mouse with PNA Menus

1. Click **Utility**
2. then **System**
3. then **Configure**
4. then **Millimeter Module Config**

There are NO programming commands to configure mmWave modules and test set.

Millimeter Module Configuration

Available Configuration(s):

Standard PNA
Config0

Selected Configuration: Config0

Test Set Properties

Selected Test Set: N5262A

Route PNA RF to rear panel *SW SRC OUT*

Enable Test Set RF ALC Mixer Mode

Max Power Limit: 11.00 dBm At Multiplier RF IN

RF Cable Loss (DO NOT include test set gain)

Power Offset: 0.00 dB Power Slope: 0.113 dB/GHz

Frequency Settings

	Start Frequency	Stop Frequency	Multiplier	Source
Multiplier RF IN:	12.5000000000 GHz	18.3333333333 GHz	6	PNA RF Source ...
Multiplier LO IN:	9.3750000000 GHz	13.7500000000 GHz	8	PNA LO Source ...
Test Port Frequency:	75.000000000 GHz	110.000000000 GHz		

Clicking OK will preset

OK Cancel Help

Millimeter Module Configuration dialog box help

Note: To protect your mmWave modules from damage, settings on this dialog can **ONLY** be changed manually. They can **NOT** be reset or changed by performing a Preset, or by recalling an Instrument State, or by a remote program.

Available Configurations Lists the Standard PNA configuration and other mmWave configurations that you have created.

- Click **New** for first-time use.
- For the N5251A, select **Broadband 10MHz - 110GHz**.

- Select **Standard PNA** to exit mmWave module operation.

Selected Configuration Shows the currently selected configuration. Edit this field to change the configuration name. Type a unique name using only alphanumeric characters and underscore.

New Click to create a new Millimeter Module configuration. A name is automatically selected. Edit the Selected Configuration field to change the configuration name.

Remove Deletes a Millimeter Module Configuration.

Test Set Properties

Selected Test Set Select a test set to use in the current configuration. The firmware does NOT check to ensure that the selected test set is connected.

Route PNA RF to rear panel "SW SRC OUT" Available ONLY on PNA-X with [option 224 or 423](#) AND when using N5261A and N5262A test sets.

When checked, Port 1 source is switched to J11 and Port 3 source is switched to J8 on the [PNA rear panel](#). Use this configuration to quickly switch the RF Output back to the PNA front panel.

Mixer Mode Check to allow mixer testing using [SMC](#). [Learn more](#).

Enable Test Set RF ALC Available for N5261A and N5262A ONLY. When checked, power is automatically leveled at the mmWave module RF input when using the standard cables and making non-pulsed measurements. Clear this box to use non-standard cables or when making pulse measurements. When cleared, the following fields become available:

Max Power Limit The maximum mmWave module RF input is limited to this value when **Test Set RF ALC** is OFF. When you exit this dialog box using **OK**, set the power out of the PNA using the [Power and Attenuator dialog](#).

Power Offset Sets the loss of the cables. The mmWave module RF input is adjusted by this amount. Positive offset increases the power.

Power Slope Helps compensate for cable and test fixture power losses at increased frequency. The mmWave module RF input power increases as the sweep frequency increases in dB/GHz. The slope is defined relative to the mmWave module RF input frequency. The slope starts at 0Hz and a positive slope will increase the power level. Range is +/- 2 dB/GHz.

Frequency Settings

Multiplier RF IN RF Frequency Range (displayed in grey fields) multiplied by this value = test port frequency range.

Multiplier LO IN LO Frequency Range (displayed in grey field) multiplied by this value plus

the IF frequency equals the test port frequency. The IF frequency is:

- 'C' Models = 8.333 MHz
- PNA-X models = 7.605 MHz

Test Port Frequency Set the Start and Stop frequencies of the selected configuration at the test ports. This becomes the displayed Start and Stop frequency of the PNA.

Important Notes

- To set Test Port Frequency, first set the appropriate **Multiplier** values that are specified in your mmWave module documentation.
- Ensure that the RF and LO Frequencies (highlighted below) are within the frequency range of the sources. The PNA offers no warning if they are NOT.

	Start Frequency	Stop Frequency
Multiplier RF IN:	12.500000000 GHz	18.333333333 GHz
Multiplier LO IN:	9.375000000 GHz	13.750000000 GHz

Source

Click a button to launch the [External Devices dialog](#) where you can select an internal or external source to be used for the PNA LO source or PNA RF source.

Cancel Closes dialog box without saving changes.

OK Saves the configuration and the **PNA is Preset** before making the appropriate settings.

Mixer Mode

Mixer measurements can be made at mmWave frequencies using [SMC](#). (VMC measurements are NOT supported.)

Beginning with A.09.40, mixer measurements can be made with a 2-port test set connected to a 4-port PNA-X. This configuration yields a 2-port mmWave system. [Learn about 2-port system connections and limitations.](#)

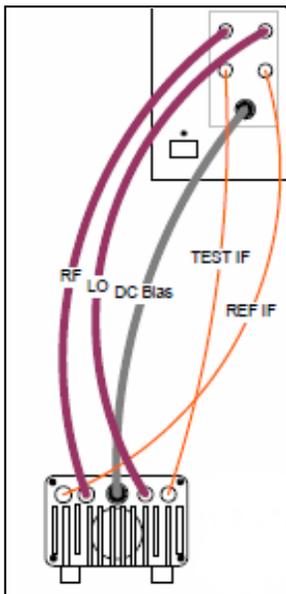
Before A.09.40, The Mixer Mode checkbox could be enabled ONLY when the number of PNA test ports matched the number of ports on the mmWave test set. This means that an N5261A (2 port test set) could ONLY be connected to a 2-port PNA and an N5262A (4 -port test set) could ONLY be connected to a 4-port PNA.

Procedure

1. Connect your DUT to the mmWave system as described below.
2. Configure this dialog ([Millimeter Module Configuration](#)). Check **Mixer Mode**, then press **OK**. This presets the PNA.
3. [Create an SMC measurement](#).
4. [Make mixer settings](#). As with standard SMC measurements, only two DUT ports can be swept in frequency. The remaining DUT port must be a fixed frequency. [See configuration used for harmonic mixers](#).
5. [Increase power](#) for mmWave modules that are connected directly to a PNA port or external source.
6. Calibrate using the [SMC Calibration Wizard with mmWave Power Control](#).

Hardware Connections for Mixer mode

The following image shows the standard connections from a N5261A or N5262A test set port to an OML mmWave module.



mmWave mixers usually require that two of the three mixer ports operate at mmWave frequencies. When **Mixer Mode** is checked on this dialog ([Millimeter Module Configuration](#)), the following restrictions apply:

- On a 2 port mmWave system, only **port 1** of the test set can be used as a mmWave frequency port. Port 2 can NOT be used.
- On a 4 port mmWave system, only **ports 1 and 3** of the test set can be used as mmWave frequency ports. Ports 2 and 4 can NOT be used.
- The [SMC parameter](#) being measured must be within the frequency range of the PNA or within the frequency

range of the banded mmWave module. Frequencies in between these ranges are allowed by the [SMC mixer setup dialog](#), but the measurement results on the screen are NOT accurate.

Connections with a 4-port mmWave system

Upconverters

- DUT Input - Connect to PNA port 2 or port 4.
- DUT LO - Connect mmWave module to test set Port 3.
- DUT Output - Connect mmWave module to test set Port 1.

Downconverters

- DUT Input - Connect mmWave module to test set Port 1.
- DUT LO - Connect mmWave module to test set Port 3.
- DUT Output - Connect to PNA port 2 or port 4.

Connections with a 2-port mmWave system

Although supported, testing mmWave mixers with a 2-port system can be challenging for the following reasons:

- Testing mmWave mixers requires that two of the three DUT ports be at mmWave frequencies.
- Only test set port 1 is capable of adequately driving a mmWave module when used as a receiver.
- Therefore, the second DUT port that requires mmWave frequencies must have the mmWave module connected directly to an external source or a PNA second source.
- When using the mmWave module as a source, only the DC Bias and RF cable is necessary. The LO cable to the mmWave module is NOT used. This is because the RF input frequencies are multiplied in the mmWave module to provide the source frequencies. So a mmWave module used as a source can use the RF cable to connect directly to the PNA second source or an external source. About +5 dBm of RF power is required to adequately drive the mmWave module.

Downconverters - requires two mmWave modules as **sources**

- DUT Input - Connect the mmWave module to the test set port 1.
- DUT LO - Connect the RF cable of the mmWave module to an external source or the PNA (SRC2) second source.
- DUT Output - Connect to PNA port 2.

Upconverters - requires a mmWave module as a **source** at the DUT LO and a mmWave module as a

receiver at the DUT Output:

- DUT Input - Connect to PNA port 2.
- DUT LO - Connect the RF cable of the mmWave module to an external source or the PNA (SRC2) second source.
- DUT Output - Connect the mmWave module to the test set port 1.

Measuring Harmonic Mixers

Harmonic mixers have a multiplier circuit in the LO port of the DUT. Enter the multiplier value in the numerator of the X LO port in the [SMC mixer setup dialog](#). This will provide the correct LO frequencies out of the appropriate source.

mmWave Module Power Level Control

Beginning with A.09.40, the following TWO features are integrated into [Guided Cal](#):

- [For S-parameter Cal](#) - Use Multiple Sensors
- [For SMC Cal](#) - Power Table

The following table shows features that can be used to provide leveled power to the input of your DUT for S-parameter and SMC measurements.

Feature	Description	Use when...	Use for...	Access the feature...
Receiver leveling	Provides a sweep-to-sweep leveled power.	Works anytime.	S-params and SMC	Before or after Cal
Use Multiple Sensors	Allows several power sensors to be used to calibrate source power.	You require more than one power sensor to complete the source power calibration of the measurement frequency range.	S-params	During Guided Power Cal
Power Table	Build or use a file that contains data of mmWave module	A power sensor is NOT available for calibration of the mmWave modules	S-params	During Std Source Power Cal

	output power vs frequency.	being used.	SMC	During Guided Power Cal
Calibrate the source at multiple power levels	Source power is measured using the specified power meter/sensor or PNA receiver to construct a 2D power table:	A component is used in the source path which does not have NOT linear gain or loss over frequency.	S-params	During Std Source Power Cal

S-parameter measurements

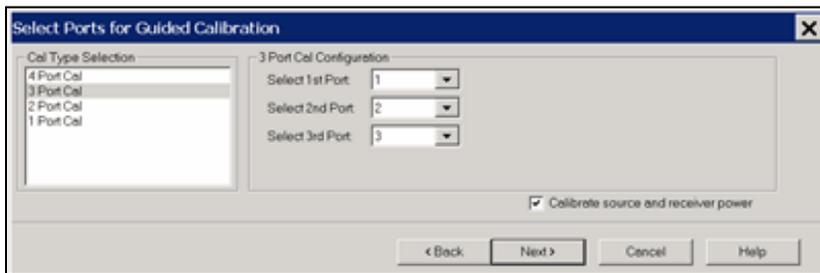
If you have one or more power sensors that spans the frequency range of your measurement, then use the following process.

Otherwise, perform a standard Source Power Cal. [Learn how.](#)

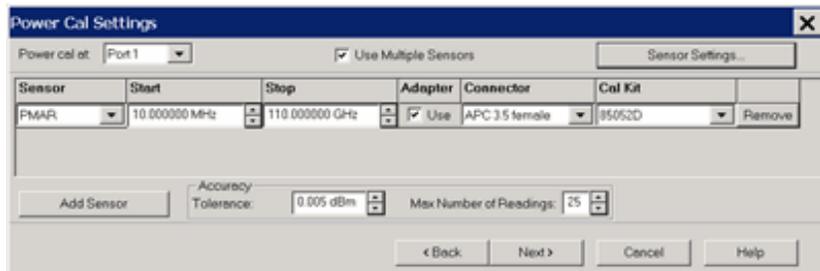
Using one or more power sensors

Check [ALC Enabled](#) (if available) on the [Millimeter Module Configuration](#) dialog.

1. With an S-parameter measurement active, press **Cal**, then **Start Cal**, then **Calibration Wizard**, then **SmartCal**.
2. On the following **Select Ports** dialog, check **Calibrate source and receiver power**, then click **Next**.



3. **Important:** In the following dialog, check **Use Multiple Sensors**, even if using only one sensor.



[Learn about this dialog.](#)

4. Complete the Guided Cal process.

Note: During the 'Connect a power sensor to port n'...step, the following error message may be displayed:

The default power level of 11 dBm is unachievable after calibration. Lower the power before starting calibration.

This means that a high amount of loss was measured in the path, and 11 dBm at the test ports will not be possible.

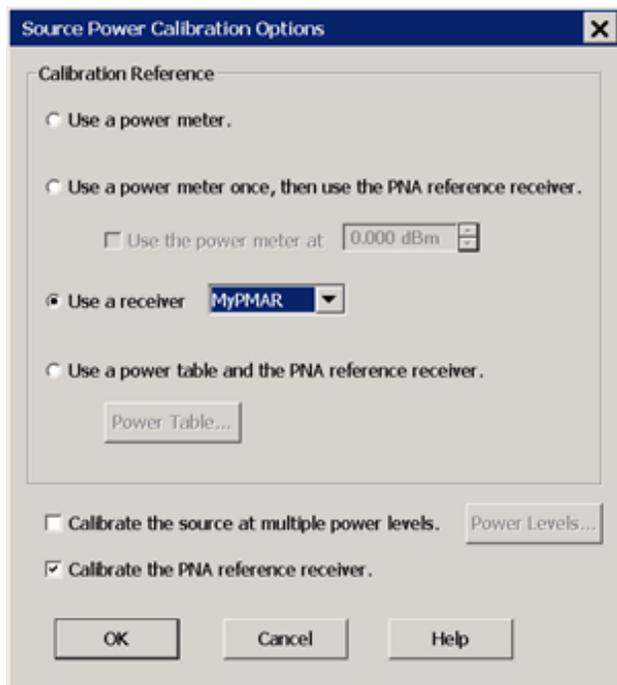
Cancel the calibration and lower the source power level using the [Power and Attenuators dialog](#).

Perform a standard Source Power Cal - S-parameter measurements

When one or more power sensors that spans the frequency range of your S-parameter measurement are NOT available, then use the following process.

Note: Perform an S-parameter calibration AFTER performing the following Source Power Cal.

1. Check **ALC Enabled** (if available) on the [Millimeter Module Configuration](#) dialog.
2. Press **Cal**, then **Power Cal** then **Source Cal** then **Options** to launch the following dialog:



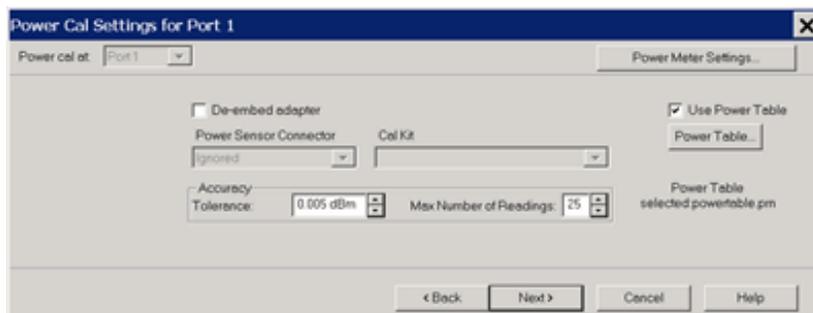
[See the help topic for this dialog](#)

3. If one does not already exist, create a power table to be used to calibrate the PNA receiver. [Learn how.](#)
4. Check **Use a power table and the PNA reference receiver.**
5. Click **Power Table**, then navigate to the *.prn file.
6. Click **OK.**
7. Check **Calibrate the source at multiple power levels.**
8. Click **Power Levels**, then enter the Max power, Min power, and Step Size at which source power should be corrected. Be sure that the source power for your measurement is within these power levels. Otherwise, source power will NOT be accurate. [Learn more about this feature.](#)
9. Check **Calibrate the PNA reference receiver**, then click **OK.**
10. On the Source Power Cal dialog, click **Take a sweep.** The output of the test set is set to Max power and a sweep is performed to calibrate the reference receiver.
11. Power is dropped for several subsequent sweeps. The calibrated reference receiver is used to fully characterize the source power.
12. The entire correction table can be saved along with the instrument state in a *.csa file. [Learn how.](#)
13. Power out of the input module should be flat and accurate.

SMC Cal

Use the following calibration process to achieve accurate, leveled power at the mmWave test ports.

1. With a configured SMC measurement active, press **Cal**, the **Start Cal**, then **Calibration Wizard.**
2. At the SMC [Calibration Setup](#) dialog, when a Thru standard is NOT available, check **Independent power cals for input and output ports (no thru).**
3. On the [Select Ports](#) dialog, check **Calibrate source and receiver power**, then click **Next.**
4. At the following Power Cal settings dialog:



[Learn about this dialog](#)

- a. When you have ONE power sensor that spans the frequency range of your SMC measurement, then click **Power Meter Settings** to configure the power sensor. There are currently NO provisions for using multiple power sensors with SMC Calibration.
 - b. Otherwise, use the following Power Table process.
 - i. If one does not already exist, create a power table to be used to calibrate the PNA receiver. [Learn how.](#)
 - ii. Check **Use Power Table**.
 - iii. Click **Power Table**, then navigate to the *.prn file. The selected *.prn file is annotated to the dialog.
 - iv. Click **OK**.
5. If you checked **Independent power cals for input and output ports (no thru)**, you will ALSO be prompted to select a power table for Port 2.
 6. Complete the [Guided SMC Cal](#) process.

Power Table

Note: This is NOT the same table that is used for the [Calibrate the source at multiple power levels](#) feature.

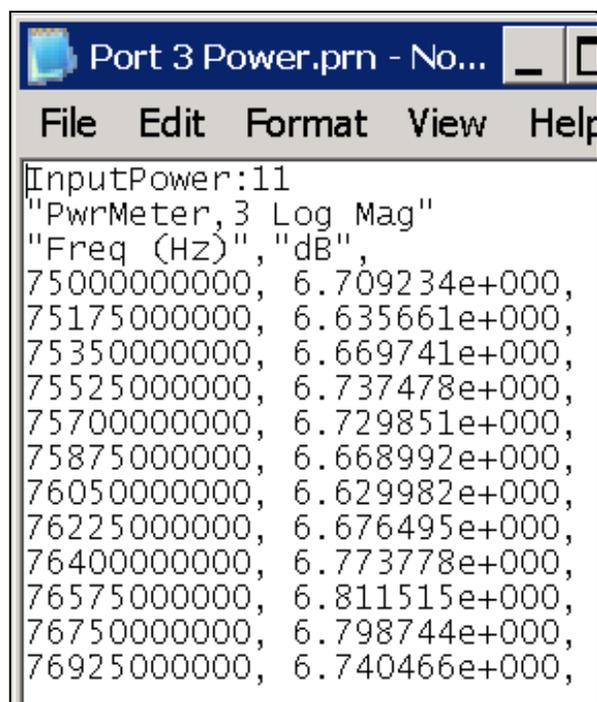
A power table is a text file with data that describes the output power of the module as a function of frequency. This is valid when the mmWave module is driven at high levels (+11 dBm). This file may have been created for you by a third party or shipped with your mmWave Module. If not, you can create this *.prn file from measured values using the following procedure:

How to create a Power Table from measured data

1. Setup the mmWave measurements including the [MM Module Configuration dialog](#) settings.
2. If a power sensor is available that spans the frequency range of your measurement, then do a source power calibration using a power meter for the first iteration. Otherwise, skip this step.
3. Set PNA power to the maximum value (+11 dBm).
4. Create an R1 trace and measure. [Learn how.](#) The loss through the MM Module is small enough to not be considered.
5. Click **File** then **Save As** to save the R1 trace to a *.prn file. Make note of the folder.
6. The file is recalled in the above procedure.

This file can now be used instead of connecting the power meter for this module.

This file can also be created manually, using a text file program such as Notepad. Copy the header information, and create the file with two columns, one for frequency and one for output power.



```
Port 3 Power.prn - No...
File Edit Format View Help
InputPower:11
"PwrMeter, 3 Log Mag"
"Freq (Hz)", "dB",
7500000000, 6.709234e+000,
7517500000, 6.635661e+000,
7535000000, 6.669741e+000,
7552500000, 6.737478e+000,
7570000000, 6.729851e+000,
7587500000, 6.668992e+000,
7605000000, 6.629982e+000,
7622500000, 6.676495e+000,
7640000000, 6.773778e+000,
7657500000, 6.811515e+000,
7675000000, 6.798744e+000,
7692500000, 6.740466e+000,
```

Example .prn file

Note: With Rev. 09.31, the first line of the *.prn file must have the Input power at which these measurements were made. Otherwise, an error message appears with the default value that will be assumed. See above image for format.

Last modified:

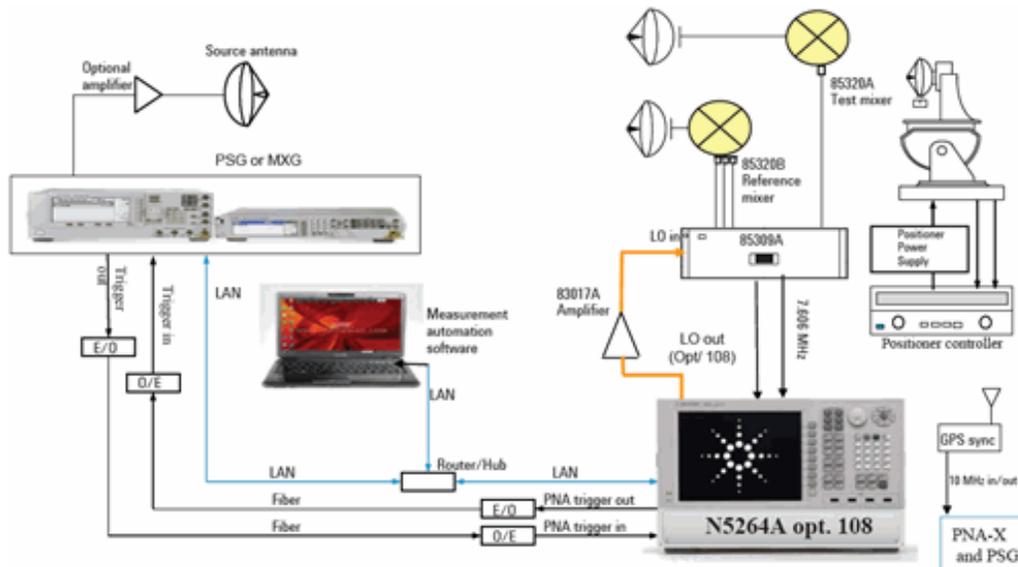
- 3-May-2012 Removed C models
- 3-Feb-2012 Added N5251A and IM Spectrum
- 20-Oct-2011 Added N5250C
- 26-Apr-2011 Added 2-port test set w/ 4-port PNA (A.09.40)
- 22-Oct-2010 New multiple power levels and prn format
- 19-Oct-2010 Added new SPC Options dialog
- 9-Dec-2009 Added Cal note for waveguide
- 28-Aug-2009 Added Mixer Mode (A.09.00)
- 11-Mar-2009 Added Leveled power (A.08.5)
- 26-Feb-2009 Added Opt 551 requirement
- 14-May-2008 Major updates for PNA-X 8.2
- 9-Apr-2007 Updated for PNA-X

9/12/06 Note - NOT compatible with Freq Offset

N5264A Measurement Receiver

The N5264A is a PNA-X with no Source Ports or Tunable receivers. This makes the N5264A a very fast and very sensitive IF receiver that has been designed specifically for antenna and radar cross-section (RCS) measurements. When used with external sources, the N5264A can make frequency-scan measurements of antennas, or make RCS measurements in time domain.

- [N5264A Options and Limitations](#)
- **Configuring the N5264A**
 - [IF Receivers](#)
 - [Internal LO Source](#) (Opt 108)
 - [External Sources](#)
 - [How to Create Measurements](#)
 - [Calibrating the N5264A](#)
- **See Also**
 - [FIFO and other Antenna Features](#)
 - [Antenna Selection Guide](#)
 - 85309 Manual updated for PNA -P/N 85310-90002
 - [Antenna and RCS Measurement Configurations](#)
 - [N5264A Specs](#)



Typical N5264A Application

Far Field Outdoor Antenna Measurements

N5264A Options and Limitations

The N5264A has the following options:

- **IF Receiver** (base model) - The A,B,C,D, and R receivers are always set at the specified IF frequency.
- **IF Receiver + LO source** (Opt 108) - In addition to the receiver-only option, this option adds a 10 MHz to 26.5 GHz LO source.
- **Fast-Sweep / FIFO Mode** (Opt 118) - These features together allow you to make very fast measurements and save the data to a remote computer. [Learn more.](#)

Limitations

The following PNA features are NOT available on the N5264A

- No Source ports
- No S-Parameters - [Arbitrary ratioed parameters ARE allowed.](#)
- [No Application support](#)
- [Limited Calibrations](#)

Configuring the N5264A

IF Receivers

- The five IF receiver inputs are on the N5264A [rear panel](#).
- All five IF receivers are measured at the same time. However, only those measurements that are displayed are updated.
- External mixers are always required to down-convert signals to the IF frequency.
- The default IF frequency for the N5264A is 7.605634 MHz.
- Change the IF frequency to any value between +14.9999 MHz and -14.9999 MHz using [SENS:IF:FREQ](#) (SCPI) or [IFFrequency](#) (COM) commands.

Internal LO Source (Option 108) - 10 MHz to 26.5 GHz

The LO source output connector is on the N5264A [rear panel](#).

The power level of the internal LO source is typically about +10 dBm and can NOT be changed.

To change the frequency of the LO Source:

1. Change the IF frequency of the measurement if necessary.
2. On the [FOM dialog](#), change **Receivers** frequencies to the RF frequency range to be measured.
3. The LO Source frequency range is set automatically. This frequency value can not be viewed.

External Sources

Because the N5264A has no internal sources (except for the optional LO), newly created measurements are displayed with a source port = 0 (zero). **Tr 1 A/R 0 LogM 10.00dB/ 0.00dB**

External sources can be configured so that they are controlled by the N5264A. [Learn how to configure an external source](#).

Once configured, all existing and new measurements are changed to use the external source as the source port. When two or more sources are configured, the first configured source is displayed by default.

The external source settings can be changed from the following dialogs:

- [Power and Attenuators dialog](#) - Controls the ON | OFF state (set to Auto by default) and Power level of the external source.
- [FOM dialog](#) - Controls the frequency range. External sources are listed by name and uncoupled by default.

How to Create Measurements

Create ratioed and unratioed measurements using the standard [Receivers tab](#).

S-parameters are not available.

Calibration

The only calibrations available are Response, Source Power and Receiver calibrations.

These are performed from the [Unguided Cal Wizard](#).

Last Modified:

27-Sep-2010	Updated Opt 108
6-Apr-2009	Removed N5242A
23-Mar-2009	Removed Fast Sweep option. Put in Antenna Features topic.
28-Aug-2008	MX New topic

Configure an External Device

Once configured (as shown in this topic), an external device will appear in, and be controlled from, relevant PNA dialogs as though it were internal to the PNA.

- [External Device Configuration dialog](#)

The following (separate) topics discuss how to set properties for these types of devices:

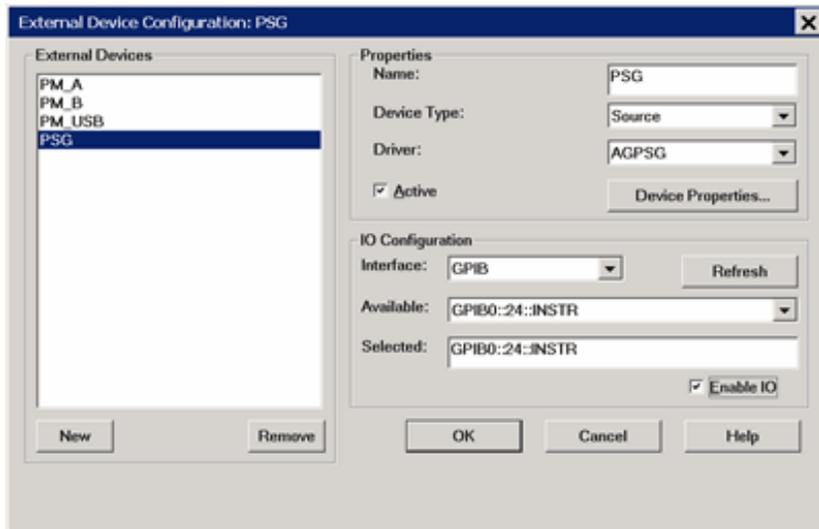
- [External Source Properties](#)
 - [Power Meter As Receiver \(PMAR\) Properties](#)
 - [External Pulse Generator Properties](#)
 - [Configure an SMU \(Source/Measure Unit\)](#)
 - [Configure a DC Source/Meter](#)
-

How to access the External Device Configuration dialog

[PNA Applications](#) have additional methods of launching this dialog.

Using front-panel HARDKEY [softkey] buttons	Using a mouse with PNA Menus
<ol style="list-style-type: none">1. Press SYSTEM2. then [Configure]3. then [External Device Configuration]	<ol style="list-style-type: none">1. Click Utility2. then System3. then Configure4. then External Device Configuration
 See Remotely Specifying a Source Port	

External Device Configuration dialog box help



Important Notes

- This dialog is used to configure the following types of external devices:
 - [External Source Properties](#) (requires FOM Opt 080)
 - [Power Meter As Receiver \(PMAR\) Properties](#)
 - [External Pulse Generator Properties](#)
 - [DC Power Analyzer](#)
 - [SMU \(Source/Measure Unit\)](#)
- To configure an external source using this dialog, your PNA must have [FOM Option 080](#). Without this option, you must control an external source manually. See [Synchronize an External Source](#) for help with manual source control.
- By default, an external device is **de-activated** when the PNA is Preset or when a Instrument State is recalled. This behavior can be changed with a [Preference setting](#) so that it remains active through a Preset or Instrument State recall.
- External Device properties are NOT saved in an Instrument State file. However, the reference to the External Device from relevant PNA dialogs IS saved. Therefore, recalling a state file that refers to a device that is NOT present will result in a “Device configuration not found” error.
- Multiple configurations for the same physical device can be Active. However, only one configuration for the same external source can have the [I/O Enabled](#).

External Devices

The devices that are currently configured appear in this list. The number of devices that can be configured is limited by the specified Interface.

New Click to create a new device configuration. The default name is Device<n>, where <n> is the next number for 'Device'.

Remove Click to remove the selected device from the list.

Properties

Name Enter a device name as it will appear when referring to this device in PNA dialog boxes. Edit the name at any time. Duplicate names are not allowed.

- Because External Devices can be used with FOM ranges, do NOT name an external device any of the following FOM range names: "primary", "receivers", or "source", "source1", source2" and so forth. [Learn more about FOM ranges](#).
- Do NOT use a parameter name such as "S11" or "R1".
- **DC Analyzer devices MUST use at least three characters in the name.**

Device Type Select one of the following:

- **DC Meter** - [Learn more](#)
- **DC Source** - [Learn more](#)
- **Power Meter (PMAR)** - [Learn more](#)
- **Pulse Generator** - Only the Agilent 81110A Pulse Generator is supported. [Learn more](#)
- **SMU** - [Learn more](#)
- **Source (RF)** - [Learn more](#)
- **None** - returned remotely before setting Device Type.

Driver Select the appropriate model to be configured.

For **Source** Device Type choose from:

(AG is short for Agilent).

- **AGESG** (ESG)
- **AGEXG** (EXG) [See configuration note](#)
- **AGPSG** (PSG)
- **AG836XX** (8360 and 8340)

- **AGMXG** (MXG) The MXG must have at least firmware A.01.44 for FOM power sweep to work correctly.
- **AGGeneric** For sources that are NOT listed but can be controlled using SCPI. Click **Device Properties**, then **Edit Commands** to send commands to these sources. [Learn how.](#)

For **SMU** Device Type, choose from:

- **B29xx** - B2900 Series
- **N67xx** - N6700 Series

All other device types have only one driver.

Active Check to make the device available for use in the relevant dialogs. An instrument state that is saved with an Active device (checked) will include the device in the state file. Otherwise, if the Active box is cleared, the device will NOT appear in the state file. [Learn more about Instrument State files.](#)

Enable I/O Clear this box to disable communication with the selected device. Do this to configure a device that is not yet connected to the PNA.

- Communication with devices is attempted when **Enable I/O** is checked, **Active** is checked, and **OK** is pressed.
- When communication is attempted, devices with **Enable I/O** checked are queried for selected limits, such as frequency, power, and number of points. If there are limit problems, the PNA sends an error and the affected channels are put into Hold. These limits are enforced by the dialog box in which they are set. Resolve the reported limit problem and then restore the triggering.
- If communication with a device is lost the affected channels are put into Hold.

Device Properties Click to launch the Properties dialog for relevant Device type:

- [Configure External Sources](#)
- [Configure a Power Meter As Receiver](#)
- [Configure an External Pulse Generator](#)
- [Configure a DC Meter / Source](#)
- [Configure a SMU](#)

IO Configuration

Interface Select the interface that is used to connect the device to the PNA. These devices will then appear in the 'Available' field. Choose from:

- **GPIB** - Devices connected to the System Controller GPIB port.
- **USB** - Devices connected to the PNA USB ports.
[See Important First-time USB connection note.](#)
- **Aliases** - Devices that are connected to ANY interface for which you created an alias. [See Configure Alias and LAN devices.](#)
- **LAN** - Devices connected to a network using a LAN connection. The PNA must also be connected to the network.

Note: Devices connected to LAN must first be configured in Agilent IO libraries before they will appear on the Available list. [See Configure Alias and LAN devices.](#)

Available Shows a list of devices that are connected to the specified IO Interface.

Refresh Click to rescan the specified interface for devices.

Selected Enter the IO configuration or select from the available list of IO Interfaces found.

Configure Alias and LAN Devices

Use this procedure to configure a device using a LAN interface. Also use for ANY device for which you want to set an alias (easily-recognized) name. The alias name appears in the Available field when Aliases is selected as the Interface.

1. On the PNA, minimize the PNA application.
2. In the system tray (lower-right corner) right-click the IO icon, then click **Agilent Connection Expert**

To Add a LAN Device:

1. In ACE, click  Add Instrument
2. Select Add LAN Instrument (TCPIP0) or USB0, then click OK.
3. Click, then enter the IP address of the external source.
4. Click Test Connection to verify communication.
5. Click **OK**.

To create an Alias for a connected device:

1. In the list of connected instruments, right click the external source, then **Add VISA Alias**
 2. Enter the same Device name that was, or will be, used in the [External Device Configuration](#) dialog.
-

Last Modified:

5-Feb-2013	Added SMU (9.90)
16-Aug-2012	Added DC analyzer name note
11-Jul-2012	Added None device type
16-Sep-2011	Removed Ext Source to separate topic. Major edits.
24-Jun-2011	Several edits - including link to first time note.
29-Apr-2011	Removed PMAR notes
10-Feb-2011	Added Guided Power Cal
13-Oct-2010	Removed PMAR name restriction
30-Jul-2009	Major changes (9.0)
7-Apr-2009	Added Dwell
23-Mar-2009	Added note about MXG and E836x
3-Sep-2008	Removed legacy content
22-Aug-2008	Added list of supported..
22-Aug-2008	Added Generic source control
11-Feb-2008	Added limitation note at top
23-Jan-2008	Added Selected ordering notes
5-Nov-2007	Added links for remote selection
18-Jul-2007	Edited for FCA LO Cal changes
30-Apr-2007	MX Modified for ALL external source config.

External Source Configuration

Once configured, an external source appears in PNA dialogs as though it were an internal source. This capability requires FOM Option 080.

In this topic:

- [How to Configure an External Source](#)
 - [Important Notes](#)
 - [Trigger Settings and Physical Connection diagrams](#)
 - [Generic Source Commands dialog](#)
-

How to Configure an External Source

1. **Important:** Create an External Source device by name (one-time). [Learn how.](#) (Separate topic)
2. Then click **Device Properties** to [Configure the External Source](#). (This topic)

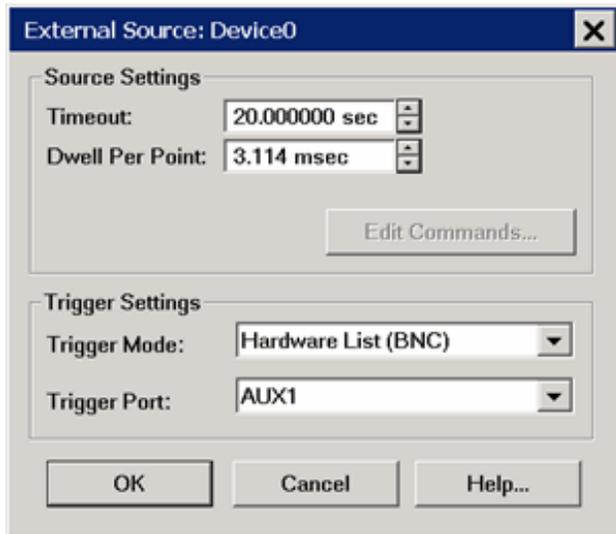
[PNA Applications](#) have additional methods of launching this dialog.

Using front-panel HARDKEY [softkey] buttons	Using a mouse with PNA Menus
<ol style="list-style-type: none">1. Press SYSTEM2. then [Configure]3. then [External Device Configuration]	<ol style="list-style-type: none">1. Click Utility2. then System3. then Configure4. then External Device Configuration



See [Remotely Specifying a Source Port](#)

External Source Configuration dialog box help



This dialog box is used to make external source settings.

Important Notes about External Sources

- First create an External Source (device) by name (one-time). [Learn how.](#)(Separate topic)
- Once you create and activate an external source from the [Configure an External Device](#) dialog, it becomes available from the following PNA dialog boxes as well as the softkeys and entry toolbar, as if it were an internal PNA source.
Use the following dialogs to set the state, frequency, and power level of the external RF source:
 - [Power and Attenuators dialog](#)
 - [FOM dialog](#)
 - [New Trace / Receivers tab dialog](#)
- By default, an external source is **de-activated** when the PNA is Preset or when a Instrument State is recalled. This behavior can be changed with a [Preference setting](#).
- External Agilent sources are usually limited to 1601 points with List-sweep mode. To 'work around' this limitation, divide the measurement among multiple channels. For example, to attain a sweep of 3200 points, create two channels of 1600 points. You can also use manual source control which supports Step-sweep mode. In this mode an external source can have up to 65,535 points. See [Synchronize an External Source](#) for help with manual source control.
- External sources should always share the same [10 MHz Reference signal](#) as the PNA. Connect a BNC cable from the PNA 10 MHz Ref Output to the External Source Input.
- All newly-activated sources are preset, with source power OFF. Source power must be turned ON in

the [Power dialog](#). Frequency Offset must be enabled in the [FOM dialog](#).

- The same source can NOT be used more than once in the same channel.
- The PNA automatically controls all trigger settings for the external source.
- [See EXG Sources configuration note](#).

Source Settings

Timeout (sec) Sets a time limit for the source to make contact with the PNA. If this time limit is exceeded, the PNA stops the measurement procedure and displays the following error message.

EXECUTION ERROR;OPC QUERY TIMEOUT ERROR: FREQUENCY NOT SETTLED

If this occurs, check the connections between your PNA and external source.

Dwell per point (ms) Applies a dwell in Hardware List triggering ONLY. Set the time (in milliseconds) the external source will wait before data acquisition.

Edit Commands Provides a method to send SCPI commands to **AGGeneric** (not listed) sources.

Trigger Settings and Physical Connection diagrams

Note: The PNA controls ALL external source trigger settings automatically (except for those on this dialog). All settings in the [External Trigger](#) dialog are ignored.

Trigger Mode

Software CW (GPIB) Slowest method.

- The external source receives each CW frequency from the PNA over GPIB, USB, or LAN. No other trigger cables are required. Although a Trigger Port selection may be available, it is NOT used.

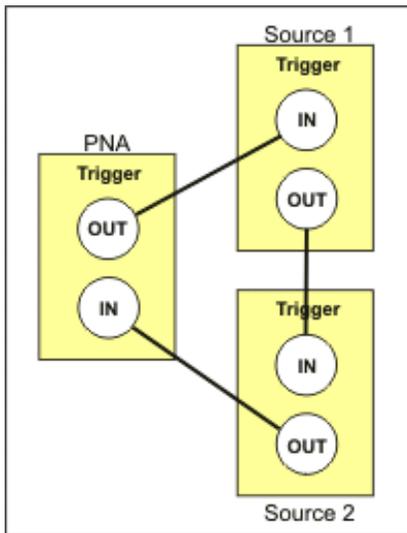
Hardware List (BNC) Fastest method.

- NOT available for AGGeneric (not listed) sources.
- The external source receives a list of CW frequencies from the PNA, then receives trigger signals through a rear-panel connector when appropriate from the PNA.
- If the number of data points used in the measurement exceeds the capability of the external source, the PNA automatically switches to Software CW (GPIB) trigger mode. This will slow the measurement significantly.

- When daisy-chaining multiple sources, the source to receive the Trigger signal from the PNA must be the first source listed in the **Selected** column of the External Device Configuration dialog. Devices are listed in the order in which they are created. You may have to delete, then re-create a source to move it down on the list.

Trigger Port Used ONLY for Hardware List Trigger Mode. Select the PNA rear panel connector to be used for triggering. The sources must be connected as follows:

- For ONE external source connect directly using AUX 1 or AUX 2 trigger pairs. [See rear panel Aux connectors.](#)
- For multiple sources, connect using the following daisy-chain image.

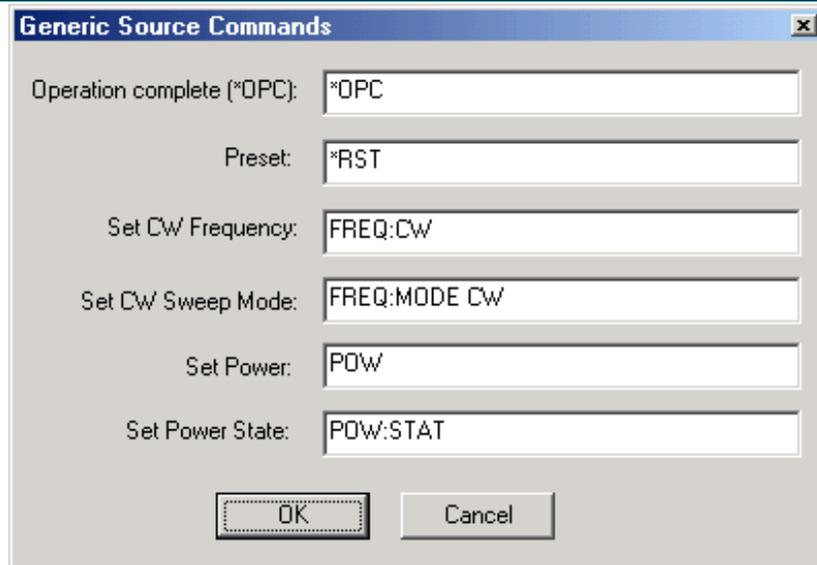


Notes

- Source 1, which receives the trigger out of the PNA, must be the first source listed on the [External Devices Configuration](#) dialog box. Devices are listed in the order in which they are created. You may have to delete, then re-create a source to move it down on the list.
- Configure/translate **EXG sources** as follows:

EXG rear-panel label	For the PNA:
Trig 1	Trig In
Trig 2	Trig Out

Generic Source Commands dialog box help



Generic Source Commands

Operation complete (*OPC): *OPC

Preset: *RST

Set CW Frequency: FREQ:CW

Set CW Sweep Mode: FREQ:MODE CW

Set Power: POW

Set Power State: POW:STAT

OK Cancel

Enter the SCPI commands that control the following functions on your AGGeneric (not listed) source. A field without a SCPI command entered will be ignored and that function will not be set.

To launch this dialog, click **Edit Commands** on the [External Source Properties](#) dialog.

Operation Complete (*OPC) .

Preset Presets the source

Set CW Frequency Sets CW Frequency

Set CW Sweep Mode Sets source sweep mode

Set Power Sets source power

Set Power State Turns power ON or OFF

9-Apr-2012	Updated 10MHz link and Removed old models
16-Sep-2011	Moved back to separate topic
24-Jun-2011	Several edits - including link to first time note.
29-Apr-2011	Removed PMAR notes
10-Feb-2011	Added Guided Power Cal
13-Oct-2010	Removed PMAR name restriction
30-Jul-2009	Major changes (9.0)
7-Apr-2009	Added Dwell
23-Mar-2009	Added note about MXG and E836x
3-Sep-2008	Removed legacy content
22-Aug-2008	Added list of supported..
22-Aug-2008	Added Generic source control
11-Feb-2008	Added limitation note at top
23-Jan-2008	Added Selected ordering notes
5-Nov-2007	Added links for remote selection
18-Jul-2007	Edited for FCA LO Cal changes
30-Apr-2007	MX Modified for ALL external source config.

Configure DC Sources and DC Meters

Once configured, one or more DC Sources and DC Meters can be controlled by the PNA. DC Power Analyzers are also supported, but they must be configured as a separate Source and Meter.

The Agilent [N6700 series](#) and [B2900 series](#) DC Analyzers are supported with configuration files that can be loaded on the DC Meter and DC Source property page. Once loaded, the SCPI commands that control the DC device can be modified and saved. [Learn how](#).

How to Configure a DC Meter or DC Source

1. **Important:** Create a DC Source / Meter device by name (one-time). [Learn how \(separate topic\)](#).
2. On the Configure an External Device dialog, click **Device Properties**. (This topic).

Using front-panel HARDKEY [softkey] buttons	Using a mouse with PNA Menus
<ol style="list-style-type: none">1. Press SYSTEM2. then [Configure]3. then [External Device Configuration]	<ol style="list-style-type: none">1. Click Utility2. then System3. then Configure4. then External Device Configuration

Once configured, set the DC source voltage and display DC meter measurements:

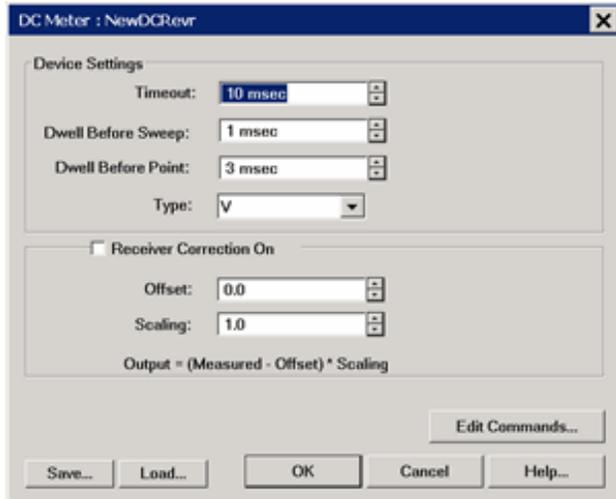
- **DC Sources:** DC Source control is available in Standard, [Gain Compression/GCX](#) , and [FCA](#) channels.
 - Set the Start and Stop voltage on the [DC Control dialog](#). To access this dialog: Press **Sweep**, then **More**, then **DC Source**.
- **DC Meters:** DC meter measurements are available in Standard, [Gain Compression /GCX](#), [Swept IMD / IMDX](#), and [FCA](#) channels.
 - In [Gain Compression /GCX](#), [Swept IMD / IMDX](#), and [FCA](#) channels, display DC parameters as you would an RF parameter, by clicking **Trace/Chan**, then **New Trace**.
 - In a Standard channel, configure an unratiod measurement. Press **Meas**, then **More**, then **Receivers**. For Numerator, select the external (or internal) DC meter.

- Change the X-axis to display the DC Meter parameters, click **Response**, then **Display**, then **Labels**, then **Select X-Axis**, then select the DC Meter.

Programming Commands

DC Source / Meter Configuration dialog box help

The DC Source and DC Meter properties are almost identical in how they operate. Both are documented here.



Device Settings

Timeout - Sets a time limit for the DC source or meter to make contact with the PNA. If this time limit is exceeded, the PNA stops the measurement and displays the following error message.

EXECUTION ERROR;OPC QUERY TIMEOUT ERROR

If this occurs, check the connections between your PNA and external device.

Dwell Before Sweep Wait time before making a sweep.

Dwell After Point

- **DC source** Wait time after setting the voltage/current at each data point.
- **DC meter** Wait time before measuring voltage/current at each data point.

Type: This setting changes ONLY the units that are displayed in the [DC Source dialog](#) and the X-axis display annotation. Use these settings with Receiver or Source Correction (Scaling and Offset) to display and scale measurements with these units. Choose from:

V (volts - default)	dBm	F (degrees)
A (amperes)	W (watts)	C (degrees)
		K (kelvin)

Note: To change the X-axis to display the DC Meter units, click **Response**, then **Display**, then **Labels**, then **Select X-Axis**, then select the DC Meter.

Receiver / Source Correction

- For a **DC source**, use the correction settings to scale and offset the output voltage.
- For a **DC Meter** (receiver), use the correction settings, along with Type, to display and scale measurements with appropriate units. For example:

Measure the voltage across a 5 ohm resistor, then display the results in A(mperes).

Using ohms law, $I = V / 5 \text{ ohms}$ or $I = V *.2$

For receiver correction, enter Scaling = .2; Offset = 0.

ON Check to apply the following correction factors to each measurement.

Offset: Enter the value to offset the DC Meter reading or set the DC Source voltage.

Scaling: Enter the value to scale the DC Meter reading or set the DC Source voltage.

Displayed Output = (Measured / Set value - Offset) * Scaling value.

Edit Commands - Click to start the Edit Commands dialog.

Important Note:

The Edit Commands dialogs (see below) **MUST** be completed. They are used to set the SCPI commands with which the PNA communicates with the DC device.

These commands are saved, along with other configuration settings, to configuration (*.xml) files. These files can then be loaded later when communicating with the same DC Device.

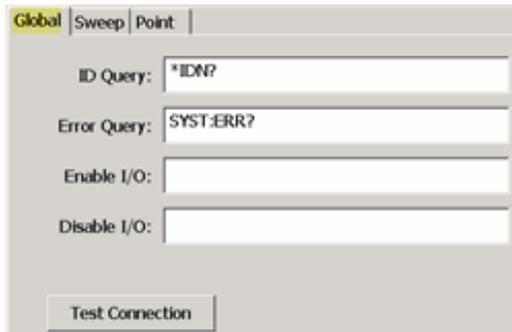
Configuration files for the Agilent N67xx and B29xx Power Analyzers are pre-loaded on the PNA. Click **Load**, then navigate to: c:\users\public\network analyzer\documents\drivers.

Save - Press to save the current DC Source or DC Meter configuration to an *.xml file. The list of files is NOT filtered by "DCMeter" or "DCSource", so **use a descriptive filename**.

Load - Press to load an existing configuration.

DC Meter Edit Commands dialog box help

Global Tab



The Global tab includes the system settings for the DC Meter.

ID Query - Enter the SCPI command to return the ID string of the DC Meter. Typically ***IDN?**

Error Query - Enter the SCPI command that is used to return DC Meter errors. Typically **SYST:ERR?**

Enable I/O - Enter the SCPI commands that is used to enable the DC Meter to read voltages.

Disable I/O - Enter the SCPI commands that is used to disable the DC Meter from reading voltages.

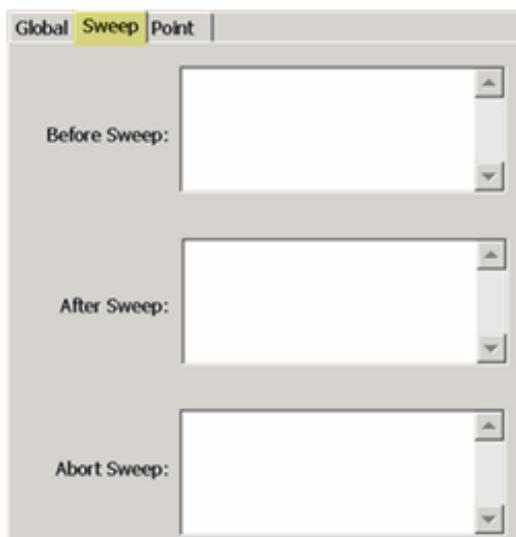
Test Connection

Click to start the Test Connection dialog. You must first have entered the I/O Configuration settings and select Enable IO on the [External Device dialog](#).



Enter a SCPI command, then click **Send** or **Send&Read** when a return value is expected.

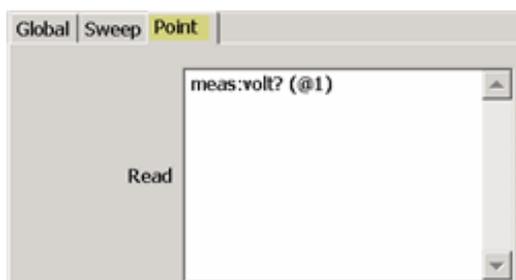
Sweep Tab



Use the Sweep Tab to send SCPI commands to the DC Meter at the beginning or end of each sweep.

Abort Sweep - Enter the SCPI command that is used to Abort or reset the DC Meter. This would be necessary when the PNA sweep is aborted or terminated. The PNA will then send the command to the DC Meter.

Point Tab



Read (commands) - Enter the SCPI command that is used to make a DC measurement at each data point.

Set (commands) - Use <% variable> to make a DC Meter setting.

Global Tab

The screenshot shows a software interface with three tabs: 'Global', 'Sweep', and 'Point'. The 'Global' tab is active. It contains several input fields and sections:

- ID Query:** *IDN?
- Error Query:** SYST:ERR?
- Enable I/O:** OUTP ON, (@1)
- Disable I/O:** OUTP OFF, (@1)
- Maximum DC Output:**
 - Read Max Using
 - Define Max As: 10.000 V
- Minimum DC Output:**
 - Read Min Using
 - Define Min As: -10.000 V
- Test Connection** button

The Global tab includes the system settings for the DC Source.

ID Query - Enter the SCPI command to return the ID string of the DC Source. Typically ***IDN?**
This entry can be left blank.

Error Query - Enter the SCPI command that is used to return DC Source errors. Typically **SYST:ERR?**

Enable I/O - Enter the SCPI commands that is used to enable the DC Source to output voltages.

Disable I/O - Enter the SCPI commands that is used to disable the DC Source from outputting voltages.

Maximum / Minimum DC Output

Read Max / Min Using - Select, then enter the commands used to return the output limits of the DC source.

Define Max / Min As - If the DC Source has no commands to return these values, or you would rather define the limit for your DC Source, select then enter the Max and Min voltage limits.

Test Connection

Click to start the Test Connection dialog. You must first have entered the I/O Configuration settings and select Enable IO on the [External Device dialog](#).



Enter a SCPI command, then click **Send** or **Send&Read** when a return value is expected.

Sweep Tab

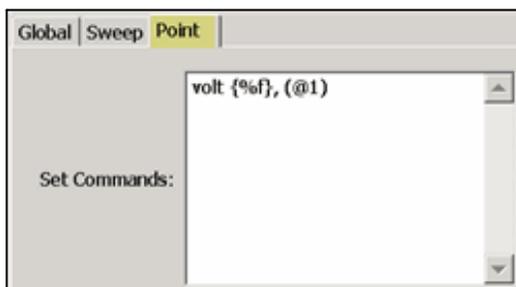


Use the Sweep Tab to send SCPI commands to the DC Source at the beginning or end of each sweep.

Typically, you might send the output ON at the beginning of each sweep, and output OFF at the end of each sweep as shown above.

Abort Sweep - Enter the SCPI command that is used to Abort or reset the DC Source. This would be necessary when the PNA sweep is aborted or terminated. The PNA will then send the command to the DC Source.

Point Tab



Note: The DC Source output voltages are configured on the [DC Source dialog](#).

This dialog is used to configure the commands that are used to communicate with the DC Source.

Set commands - Enter the SCPI command, enclosed in {curly brackets} to output (set) a voltage/current from the DC Source for each data point.

- **{%f}** - The value is a double value. (Most common).
- **{%d}** - The value is a integer. This would be used when the voltage controls a remote switch. For example, you can program the value to: "0,1,0,1,0,1....". where "0" = OFF and "1" = ON.

Last modified:

30-Oct-2012	Modified DC parameter access
3-Aug-2012	Added DC parameter access
20-Sep-2011	New topic

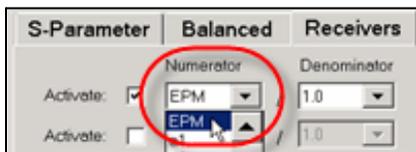
Configure a Power Meter As Receiver (PMAR)

When a power meter is configured as a PNA receiver (in [standard measurement channels](#) ONLY), you can...

- Extend the number of measurement receivers.
- Use the power meter as a scalar detector.
- Monitor the power at any point in a measurement system.
- Use multiple power meters in a [Guided Power Cal](#) to cover a wide frequency range.
- Use the power meter to level the stimulus power at any point in a measurement system.
- Use the power sensor as a PMAR device to confirm the accuracy of a Source Power Cal. [Learn how.](#)

Once configured, a power meter can be used like any other PNA receiver in the following dialogs:

- [New Trace / Meas dialog](#) - used in Ratioed and Unratioed measurements.



- [Receiver Leveling](#)
- [Frequency Offset Mode](#) - Extend frequencies beyond PNA

See Also

- [Supported Power Meters](#)
- [Important first-time USB connection note.](#)

How to Create and Configure a PMAR Device

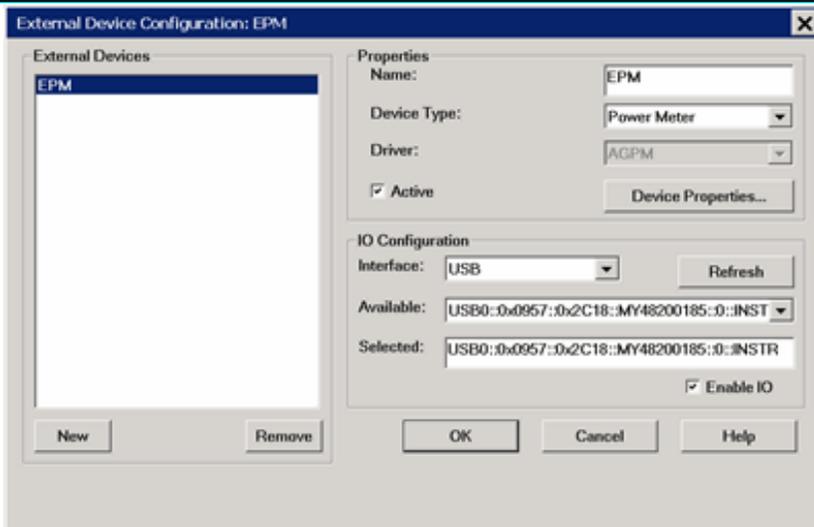
1. Create a PMAR device by name (one-time).
2. Then click **Device Properties** to [configure the Power Meter/Sensor](#).

[PNA Applications](#) have additional methods of launching this dialog.

Using front-panel HARDKEY [softkey] buttons	Using a mouse with PNA Menus
<ol style="list-style-type: none">1. Press SYSTEM2. then [Configure]3. then [External Device Configuration]	<ol style="list-style-type: none">1. Click Utility2. then System3. then Configure4. then External Device Configuration

Programming Commands

External Device Configuration dialog box help



This dialog allows you to create and configure a power meter to be used as a receiver by the PNA. Once you create and configure a power meter from this dialog box, it becomes available from PNA dialog boxes as well as the softkeys and entry toolbar, as if it were an internal PNA receiver.

- This dialog is ALSO used to configure an **External Source**. [Learn more.](#)

- To configure a single power meter for a Source Power Cal, use the [Power Meter Settings](#) dialog.

Important Notes

- By default, an external PMAR device is **de-activated** when the PNA is Preset or when an Instrument State is recalled. This behavior can be changed with a [Preference setting](#) so that it remains active through a Preset or Instrument State recall.
- PMAR configuration is NOT saved in an Instrument State file. Therefore, recalling a state file that refers to a device that has been removed, or recalling a state file on a different PNA will result in a "Device configuration not found" error.

External Devices

The devices that are currently configured appear in this list. The number of devices that can be configured is limited by the specified Interface.

New Click to create a new PMAR configuration. The default name is Device<n>, where <n> is the next number for 'Device'.

Remove Click to remove the selected device from the list.

Properties

Name Enter a device name as it will appear when referring to this device in all PNA dialog boxes. Edit the name at any time. Duplicate names are not allowed.

Notes

- Because External Devices can be used with FOM ranges, do NOT name an external device any of the following FOM range names: "primary", "receivers", or "source", "source1", source2" and so forth. [Learn more about FOM ranges](#).
- Do NOT use a parameter name, such as "S11, or "R1".

Device Type Select **Power Meter**.

Driver Use **AGPM** for all Agilent Power Meters. See [Supported Power Meters](#)

Active Check to make the device available for use in the FOM, New Trace, and Receiver Leveling dialogs. An instrument state that is saved with an Active device (checked) will include the device in the state file. Otherwise, if the Active box is cleared, the device will NOT appear in the state file.

Note: Multiple PMAR configurations for the same physical device can be Active and Enabled.

Enable I/O Clear this box to disable communication with the selected device. You would do this

to configure a device that is not yet connected to the PNA.

- Communication with devices is attempted when **Enable I/O** is checked, **Active** is checked, and **OK** is pressed.
- If communication with a device is lost, the affected channels are put into Hold.
- When communication is attempted, devices with **Enable I/O** checked are queried for limits for frequency, power, and number of points. If there are limit problems, the PNA sends an error and the affected channels are put into Hold. These limits are enforced by the dialog box in which they are set. Resolve the reported limit problem and then restore the triggering.
- Communication is also attempted when clicking the **Settings** button on the [Configure Power Sensor](#) dialog. You can not change any of the sensor settings unless **Enable I/O** and **Active** are checked and communication is possible with the sensor.

Device Properties Click to launch the [Configure Power Sensor](#) dialog.

IO Configuration

Interface Select the interface that is used to connect the device to the PNA. These devices will then appear in the 'Available' field. Choose from:

- **GPIB** - Devices connected to the System Controller GPIB port.
- **USB** - Devices connected to the PNA USB ports. [See Important First-time USB connection note.](#)
- **Aliases** - Devices that are connected to ANY interface for which you created an alias. [See Configure Alias and LAN devices.](#)
- **LAN** - Devices connected to a network using a LAN connection. The PNA must also be connected to the network. **Note:** Devices connected to LAN must first be configured in Agilent IO libraries before they will appear on the Available list. [See Configure Alias and LAN devices.](#)

Available Shows a list of devices that are connected to the specified IO Interface.

Refresh Click to rescan the specified interface for devices.

Selected Enter the IO configuration or select from the available list of IO Interfaces found.

Configure Alias and LAN Devices

Use this procedure to configure a device using a LAN interface. Also use for ANY device for which you want to set an alias (easily-recognized) name. The alias name appears in the Available field when Aliases is selected as the Interface.

1. On the PNA, minimize the PNA application.
2. In the system tray (lower-right corner) right-click the IO icon, then click **Agilent Connection Expert**

To Add a LAN Device:

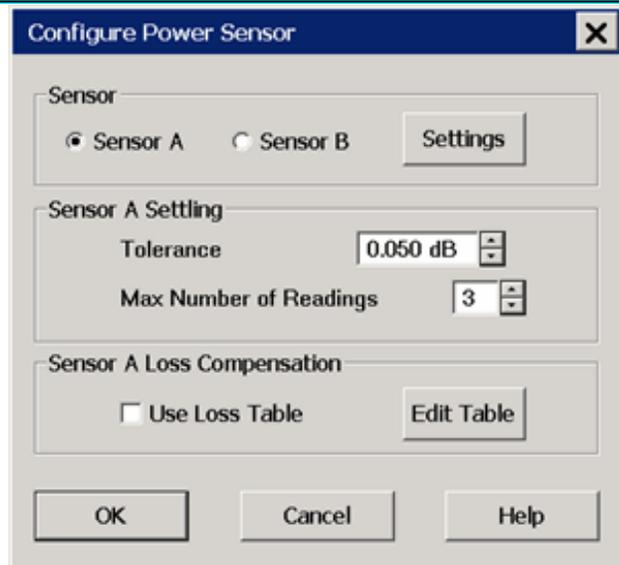
1. In ACE, click  Add Instrument
2. Select Add LAN Instrument (TCPIP0) or USB0, then click OK.
3. Click, then enter the IP address of the external source.
4. Click Test Connection to verify communication.
5. Click **OK**.

To create an Alias for a connected device:

1. In the list of connected instruments, right click the instrument, then **Add VISA Alias**.
2. Enter the same Device Name that was, or will be, used in the [External Device Configuration](#) dialog.

Power Sensor Configuration dialog box help

Programming Commands



To launch this dialog, with the PMAR device selected in the [External Device Configuration](#) dialog, click **Device Properties** .

This dialog is used to configure a power meter / sensor for use as a receiver.

To configure a single power meter for a Source Power Cal, use the [Power Meter Settings](#) dialog.

About Power Sensor Calibration

PMAR traces are NOT calibrated using standard PNA calibrations, including response corrections.

PMAR traces are calibrated using methods that are appropriate for the selected sensor. Follow the proper guidelines for zeroing or calibrating the sensors that are in use. Check to ensure that the selected sensor is appropriate for the frequency range and the power level at which PMAR measurements occur.

The PNA does not automatically prompt you to perform a calibration.

To calibrate a power sensor, click **Settings** on this dialog box, then click **Zero/Calibrate Sensor**. [Learn more](#).

Note: By default, a PMAR is de-activated when the PNA is Preset or when a Instrument State is recalled. This behavior can be changed with a [Preference setting](#).

Sensor

For power sensors that are connected to a power meter, select a sensor to configure.

Settings Click to launch the [Power Sensor Settings](#) dialog.

When pressed, communication with the sensor is tested. Sensor settings can NOT occur unless **Enable I/O** is checked on the [External Device Configuration dialog](#), and the sensor is properly connected and configured.

Sensor Settling

Each power meter reading is "settled" when either:

- two consecutive meter readings are within this Tolerance value **or**
- when the Max Number of Readings has been met.

The readings that were taken are averaged together to become the "settled" reading.

Tolerance When consecutive power meter readings are within this value of each other, then the reading is considered settled.

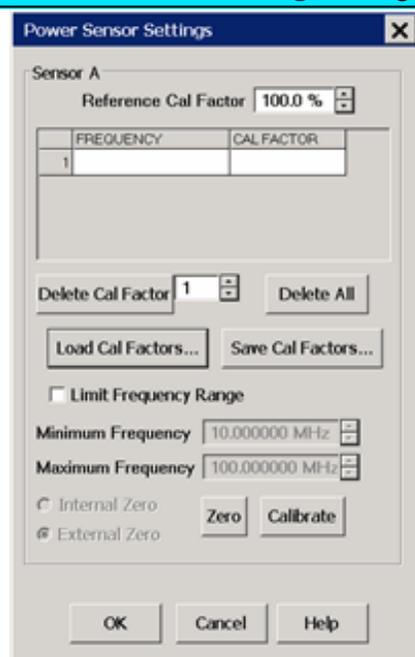
Max Number of Readings Sets the maximum number of readings the power meter will take to achieve settling.

Sensor Loss Compensation

Use Loss Table Select this checkbox to apply loss data to Source Power calibration correction (such as for an adapter on the power sensor).

Edit Table Invokes the [Power Loss Compensation](#) dialog box.

Power Sensor Settings dialog box help



This dialog appears when you click the **Settings** button on the [Configure Power Sensor](#) dialog.

Note: Be sure that the frequency range of your power sensor covers the frequency range of your measurement. This does NOT occur automatically.

Sensor A (B) Displays one of the following messages depending on type of sensor.

- **Not connected** The PNA is not detecting a power sensor.
- **Cal factors are contained within this sensor** Internal Reference Cal Factor and Cal Factor data are loaded automatically. The following table settings do not apply.
- **Sensor Data** Allows the following entries for power sensor data:

- **Reference Cal Factor** Specifies the Cal Factor for the 50 MHz reference signal.
- **Cal Factor Table** Specifies the frequency and corresponding Cal Factor for the sensor.
- **Delete Cal Factor** Deletes the indicated row in the table.
- **Delete All** Deletes all data in the table.
- **To Add a Row** to the table, click on a row in the table and press the down arrow on either the PNA front panel or keyboard. A row is added to the bottom of the table. The table is automatically sorted by frequency when OK is pressed.

Load Cal Factors Click to load cal factors from a *.csv file that you create from the cal factors that appear on the sensor. The first line of the file **MUST** have the reference Cal Factor (typically 100), followed by Freq / Cal Factor pairs as show in the following image:

	1	2
1	100	
2	50	113.3
3	51	108.9
4	52	109.1
5	53	106.5

Save Cal Factors Click to save the cal factor table to a *.csv file.

Limit Frequency Range

- Check to limit the use of the power sensor to those within the Minimum and Maximum frequency values.
- Clear to use the power sensor for all measurements. If the measurement frequency is not within the Minimum and Maximum frequency values, the closest min or max correction data is used for the measurement.

Minimum Frequency Specifies the minimum frequency range for the sensor.

Maximum Frequency Specifies the maximum frequency range for the sensor.

Zero and Calibrate the Power Sensor

For highest accuracy, Zero AND Calibrate the power sensor before measuring data. Follow prompts that may appear.

Zero - If the following settings are 'greyed', Internal or External zeroing is selected automatically based on the power meter/sensor model. Otherwise, select the appropriate type of zeroing to perform, then press **Zero**.

- **Internal Zero** - A switch inside the power sensor removes the sensor from the incident power.
- **External Zero** - Requires that you physically remove the sensor from incident power.

Note for the U2000 Series USB power sensors

Calibration is NOT available. Select External Zero ONLY when the power to be measured is **below** the specified level. Otherwise, the U2000 series performs internal zeroing automatically when needed. See your power sensor documentation for more details.

- U200xA - below -30 dBm
- U200xH - below -20 dBm
- U200xB - below 0 dBm

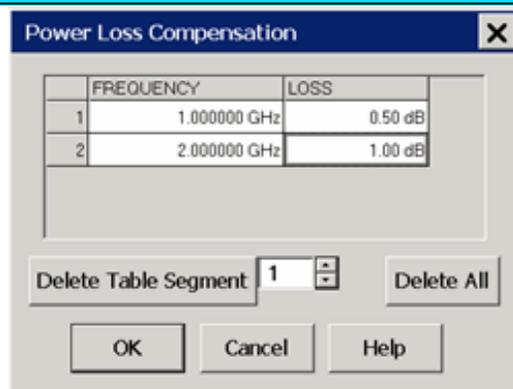
If your U2000 power sensor 'hangs' when external zeroing, upgrade the power sensor firmware to Rev. A.01.02.00 or higher to fix this problem.

Calibrate - Available when the selected sensor has calibration capability. Calibration involves measuring an internal 1 mW source.

- Agilent P-Series sensors have an internal reference so you can calibrate them without connecting to the meter's reference port.
- Agilent U2000 USB power sensors do not require calibrating.
- For other sensors, refer to the documentation to determine if it has calibration capability.

Press **Calibrate**, then follow the prompts.

Power Loss Compensation dialog box help



To Add a Row to the table, click on a row in the table and press the down arrow on either the PNA front panel or keyboard.

To Edit a value, double-click in the cell to be edited.

Compensates for losses that occur when using an adapter or coupler to connect the power sensor to the measurement port. These components will be removed when the calibration is complete. To account for components that will remain during the measurement, use the [Power Offset setting](#).

The Frequency / Loss pairs define the amount of loss for the entire frequency range. For example, using the entries in the above dialog image:

- 0.5 dB is used to compensate power sensor measurements up to 1 GHz.
- Each data point between 1 GHz to 2 GHz is linearly interpolated between 0.5 dB and 1 dB.
- 1 dB is used above 2 GHz.
- A single frequency/loss segment is applied to the entire frequency range.

Beginning with A.09.80, enter up to **9999** segments to achieve greater accuracy. Previously the limit was 100.

These values can be loaded from an S2P file using the [Characterize Adaptor Macro](#).

Note: Large segment counts with one or more power sensors can result in long load and close times for the PNA Application.

Frequency Enter a frequency in Hz.

Loss Enter a loss as a POSITIVE value in dB. To compensate for gain, use NEGATIVE values.

Delete Table Segment Deletes row indicated in the field.

Delete All Deletes all data in the table.

The Power Loss Compensation table survives PNA Preset and Power OFF. To NOT use Loss compensation, clear the Use Loss table checkbox on the [Configure Power Sensor](#) dialog.

Use a PMAR Device to confirm a Source Power Cal

[Learn how to create and configure PMAR device.](#)

After a Source Power Cal has been performed, use the same sensor as a configured PMAR to analyze the accuracy of the Calibration.

1. Create a PMAR device with the power sensor that will be used for the Source Power Cal.

2. Perform a Source Power Cal. [Learn how.](#)
 3. Create an unratioed measurement with the PMAR device. [Learn how.](#)
 4. With the power sensor still connected to the test port, monitor the corrected source power using [Min and Max markers](#) or the [Trace Statistics peak-to-peak](#) feature.
-

Last Modified:

13-Aug-2013	Added Load and Save csv file
16-Jul-2012	Added separate Zero and Cal buttons (9.70)
14-May-2012	Increased limit for Loss table
5-Apr-2012	Link to supported PM
6-Dec-2011	Added std meas class only
10-Jun-2011	Modified list of supported PMs
29-Apr-2011	Separated source and PMAR
22-Oct-2010	Added support for N991x meter
25-Aug-2009	MX New topic

Configure and Use External Pulse Generators

Once configured, one or more 81110A External Pulse Generators can be accessed from the PNA [Integrated Pulse Application](#). The external pulse generators can be used without Opt. 025 (internal pulse generators). However, the Integrated Pulse App is available ONLY with Opt. 025.

Only the 81110A Agilent Pulse Generator is supported.

In this topic:

- [How to Configure an External Pulse Generator](#)
- [Pulse Generator Configuration dialog box help](#)
- [Using External Pulse Generators with the Integrated Pulse App](#)

See Also

[Integrated Pulse Application](#)

[IF Path Configuration](#)

[81110A Quick Start Guide.](#)

How to Configure an External Pulse Generator

1. **Important:** Create an External Pulse Generator device by name (one-time). [Learn how \(separate topic\)](#).
2. On the [Configure an External Device dialog](#), click **Device Properties** (this topic).
3. Setup the external pulse generator in the [Integrated Pulse Application](#).

Using front-panel HARDKEY [softkey] buttons

1. Press **SYSTEM**
2. then **[Configure]**
3. then **[External Device Configuration]**

Using a mouse with PNA Menus

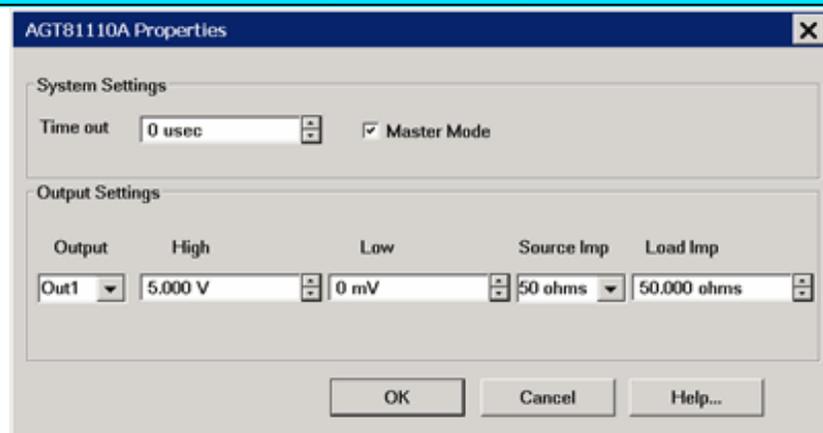
1. Click **Utility**
2. then **System**
3. then **Configure**
4. then **External Device Configuration**

[Programming Commands](#)

Tip: In the [External Device Configuration dialog](#), you can configure the same 81110A twice; once for each output module. For example:

- Name = "81110A-1" Output = **Out1**
- Name = "81110A-2" Output = **Out2**

Pulse Generator Configuration dialog box help



System Settings

Time out - Set the amount of time allowed to communicate with the external pulse generator. If communication has not been established before this amount of time has elapsed, a Timeout message will appear. Check connection settings on the [External Device dialog](#).

Master Mode - When checked, the 81110A trigger mode is set to Internal. This also causes the 81110A to appear as a selection on Integrated Pulse App, [Master Pulse Trigger](#) setting. When selected here and on that dialog, the timing of configured 'slave' pulse generators is controlled by the 81110A pulse generator. Although more than one configured pulse generator can have the Master Mode setting checked, only one pulse generator can be connected to the rear-panel Pulse connections. [Learn more about making physical connections](#).

When this setting is cleared, the 81110A trigger mode is set to External and can be configured as a 'slave' pulse generator to the PNA internal pulse generators or another external pulse generator.

Output Settings

The following are 81110A settings made by the PNA. Some settings may not be possible depending on the modules that are installed on the 81110A. Please refer to the [81110A Quick Start Guide](#) for more information.

Output - Select an output on the 81110A.

High/Low - Set the pulse voltage levels at the 81110A output.

Src Imp (Source Impedance) - Source impedance of the pulse generator output.

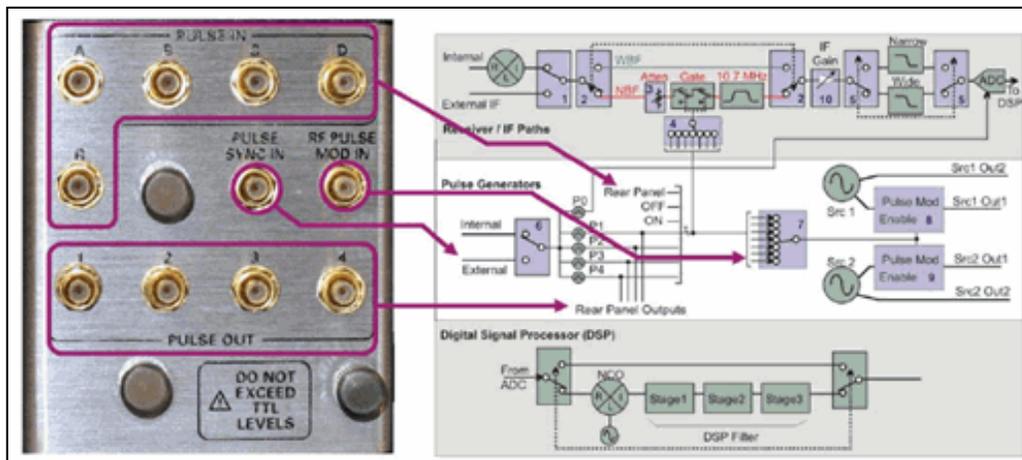
Load Imp (Load Impedance) - The load impedance value expected at the pulse generator output.

Using External Pulse Generators with the Integrated Pulse App

Once configured, an external pulse generator can be used with the Integrated Pulse App as though it were an internal pulse generator.

[N1966A Pulse I/O Adapter.](#)

See an enlarged view of the [IF Block diagram](#)



An External Pulse Generators can be used for ONE OR MORE of the following pulsed functions within the Integrated Pulse Application.

- [Modulate the sources](#)
- [Drive the IF \(Receiver\) Gates](#) (Narrowband mode ONLY).
- [Trigger the ADC](#) to make receiver measurements (Wideband mode ONLY).

How to Modulate a Source with an External Pulse Generator

When using an **external** source modulator (Z5623AH81):

- **Connect:** the 8110A to the Z5623AH81 as shown in the [Narrowband Pulse](#) topic.
- **Setting:** On the [Pulse Generator Setup](#) dialog, disable (clear) the Internal Pulse Modulators .

When using **internal** source modulators, the external pulse generator can drive the internal modulators in two ways:

- 81110A directly to the internal pulse modulators.
 - **Connect:** 81110A to RF Pulse Mod In on the N1966A OR rear-panel [Pulse I/O connector](#).
 - **Setting:** On the [Pulse Generator Setup](#) dialog, set **Modulator Drive** to "External".
- 81110A drives internal pulse generators, which drives the internal modulator.
 - **Connect:** 81110A to **Pulse Sync IN**, on the N1966A OR rear-panel [Pulse I/O connector](#).
 - **Settings:**
 - On the Pulse Generator Configuration dialog (above) check **Master mode**.
 - On the [Pulse Setup](#) dialog, set **Master Pulse Trigger** to <ext pulse gen name> .

How to Gate IF Receivers with an External Pulse Generator

(Used ONLY in Narrowband mode.)

When IF Gating is used, the external drive can be routed in two ways:

- 81110A drives gates directly at the rear-panel IF Gate inputs.
 - **Connect:** 81110A to the **Pulse IN** for one or more PNA receivers on the N1966A OR rear-panel [Pulse I/O connector](#).
 - **Setting:** On the [Pulse Setup](#) dialog, under **Measurement Timing**, for the receivers to be gated, set Pulse Gen to <ext pulse gen name>.
- 81110A drives the internal generators, which drive the gates.
 - **Connect:** 81110A to **Pulse Sync IN**, on the N1966A OR rear-panel [Pulse I/O connector](#).
 - **Settings:**
 - On the Pulse Generator Configuration dialog (above) check **Master mode**.
 - On the [Pulse Setup](#) dialog, set **Master Pulse Trigger** to <ext pulse gen name>.
 - On the [Pulse Setup](#) dialog, under **Measurement Timing**, for the receivers to be gated, set Pulse Gen to the internal pulse generator (Pulse0 through Pulse4) to be used to pulse the Rcvr<n>.

Set unique pulse Width and Delay for the Receiver.

How to trigger the ADC with an External Pulse Generator

(Used ONLY in Wideband mode).

Pulse0 is hardwired to trigger the ADC. However, the following shows how P0 be driven by the external pulse generator timing.

- **Connect:** 81110A to **Pulse Sync IN**, on the N1966A OR rear-panel [Pulse I/O connector](#).
 - **Settings:**
 - On the Pulse Generator Configuration dialog (above) check **Master mode**.
 - On the [Pulse Setup](#) dialog, set **Master Pulse Trigger** to <ext pulse gen name>.
 - On the [Pulse Setup](#) dialog, under **Measurement Timing**, for the receivers to be triggered, set Pulse Gen to **Pulse0**. Set unique pulse Width and Delay for the Receiver.
-

Last modified:

16-Sep-2011 New topic (A.09.50)

Synchronize PNA with an External (PSG) Source

Beginning with PNA Rev. 7.22, the PNA External Source Control feature can be used to automatically control external sources. However, this feature requires certain PNA options. [Learn more.](#)

Many PNA measurements require the use of at two sources. If your PNA has only one internal source, an external source is required. For example, when measuring the insertion loss of a mixer, the LO must be swept at the same time as the RF input. This requires the PNA and external source to be synchronized.

The following procedure shows how to manually synchronize the PNA with an Agilent PSG Source. Although the settings will be different, the concept is useful with other sources.

Hardware configuration

- Connect the PNA and PSG Time Base ([PNA 10 MHz OUT](#) to PSG 10 MHz IN)

Connect either pair (1 or 2) of the [AUX Trigger I/O connectors](#) as follows:

- PNA AUX Trig IN to PSG Trigger OUT
- PNA AUX Trig OUT to PSG Trigger IN

[Learn more about the AUX Trigger capabilities.](#)

PNA Settings

- [Number of points](#): **Same as PSG**
- [Frequency span](#): Does **NOT** have to be the same as PSG

[PNA Trigger Settings](#)

- Trigger Source: **Internal, Manual**
- Trigger Scope: **Channel**
- Channel Trigger State: **Same as PSG Sweep Repeat setting.**([Continuous or Single](#))
- Point Sweep: **Checked**

PNA Auxiliary Trigger Settings

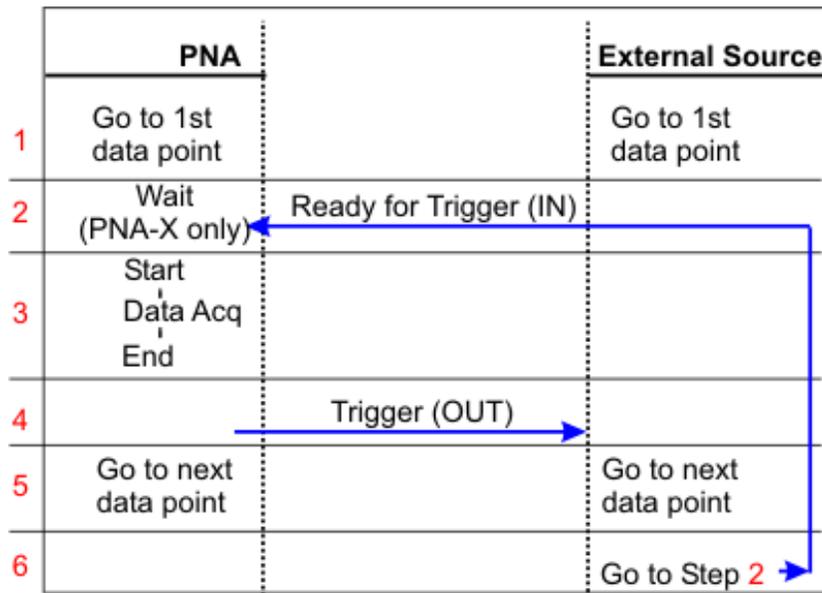
Input	
Where settings are made:	Aux Trigger Tab
Level / Edge:	Same as PSG (Hi or Low)
Accept Trigger Before Armed:	N/A
Handshake	Checked
Output	
Where settings are made:	Aux Trigger Tab
Enable	Checked
Polarity:	Same as PSG
Position:	After
Per Point	Checked

PSG Settings

- Number of points: **Same as PNA**
 - Sweep: **Step** or **List**
 - Sweep Trig: **Free Run**
 - Sweep Repeat: **Same as PNA Channel Trigger State** ([Continuous](#) or [Single](#))
 - Sweep Direction: **UP**
 - Point Trig: **Ext**
 - Manual Mode: **OFF**
 - Trigger In / Out Polarity: **Same as PNA**
-

What is Happening?

The following is a flow diagram showing the handshake / synchronization process between the PNA and an External Source.



Text Description

1. After the measurement setup is complete, both instruments wait on the first data point of a measurement sweep. Both instruments are configured for Continuous or Single sweep.
2. ([see note below](#)) A trigger signal from the source starts the measurement. This is usually accomplished by a key press on the source front panel.
3. PNA data acquisition (measurement) starts, and then stops AFTER the first data point acquisition.
4. The PNA sends a trigger signal out to the source telling it to move to the next frequency data point. This signal can optionally be sent BEFORE data acquisition if required by your application.
5. The external source and PNA move to the next data point. The source usually takes longer than the PNA.
6. The source sends the Ready for Trigger signal to the PNA. If enabled, the PNA waits indefinitely for a trigger signal from the source. The Ready for Trigger signal is latched.

Step 2 Note The PNA looks for a level trigger at the start of each sweep, and an edge thereafter. This assumes that the external source ready line will remain in the ready state (high or low) until triggered (step 4) and will then transition to the NOT ready state while moving to the next frequency, and then transition again to the ready state.

How do you know when the PNA and PSG are in synch?

The measurement results are the ultimate test of whether the source and PNA are synchronized. However, it is possible to see the PSG and PNA sweeping at exactly the same time.

First, lower the PNA [IFBandwidth](#) or increase the [sweep time](#) so that the sweep is slow enough to watch the sweep indicator moving across the PNA screen. At the same time, watch the PSG "progress bar" as it moves through the entire sweep.

If the PNA is stopped in the middle of a sweep, then retriggered, it returns to the first data point. The PSG continues from where it stopped. Therefore, to re-synch the two instruments, the PSG needs to return to the first data point. There are a number of ways to do this. One way is to press the PSG **Manual** button to ON, then OFF. Then trigger a new sweep.

To trigger a sweep

- **Single** Trigger mode: Both the PNA and PSG Single (trigger) buttons must be pressed (in any order) for each trigger.
- **Continuous** Trigger mode: First, reset the PSG to the first data point, then press the PNA Continuous (trigger) button.

Maintaining Synchronization

In general, the above setup should start the two instruments sweeping simultaneously. However, any interaction with the PNA could cause the PNA sweep to abort or delay, in which case the two instruments will be out of sync. To avoid this, you can use the PNA [Interface Control](#) feature to send an ABORT to the external device after each sweep.

When the PNA ends a sweep, it sends an ABORT to stop the source. A trigger signal is then sent, either Continuous (automatically) or Single (manual). In either case, both instruments start sweeping in synch.

This takes more time to sweep, but maintains synchronization.

For example, to use this feature with Agilent's PSG source, you would add the following:

On the "After Sweep End" tab, type:

```
24 :ABORT
```

Where 24 is the GPIB address of the source.

Last Modified:

12-Apr-2012	Removed references to old models
21-Dec-2010	Fixed source model number
24-Aug-2010	Fixed PSG trigger labels
11-Feb-2008	Updated note
1-Jun-2007	Added 7.22 update
1-Jan-2007	MX Updated for PNA-X

E5091 Test Set Control

The E5091A is a popular Agilent Technologies 7-port / 9-port test set. Although the test set was originally designed to work with the ENA Network Analyzer, it also works well with the PNA. This topic describes how to control the test set from the PNA. For more information about the test set, refer to your E5091A documentation.

- [Overview](#)
- [Connecting the E5091A](#)
- [How to make E5091A test set Control Settings](#)
- [Calibrating with the E5091A](#)

Other System Configuration Topics

Overview

When connected to the PNA, the E5091A test set provides full 7-port or 9-port test capability. The E5091A can be configured to switch a different test set path for each PNA channel. When all channels have been configured, the entire measurement setup and calibration can be [saved to a .cst or .csa file](#) to be recalled later. In addition, the [Channel Settings Table](#) that is appended to a printed hardcopy of a measurement includes the E5901A Port Control settings.

Notes:

- ONLY the 7-port and 9-port test sets are supported with the PNA.
- Works with all 4-port PNA models.
- The E5091A test set has a maximum useful frequency of 11 GHz.
- The E5091A test set Control can be automated using [SCPI](#) and [COM](#) commands.
- When [enabled](#), a second status bar row appears which indicates the test set that is being controlled and the current switch state.
- Test set path switching occurs just before a channel is triggered. If a [channel trigger state is Hold](#), switching for that channel does not occur.
- PNA sweep speed will be slightly slower when using the E5091A to switch measurement paths.

Connect and Configure the E5091A

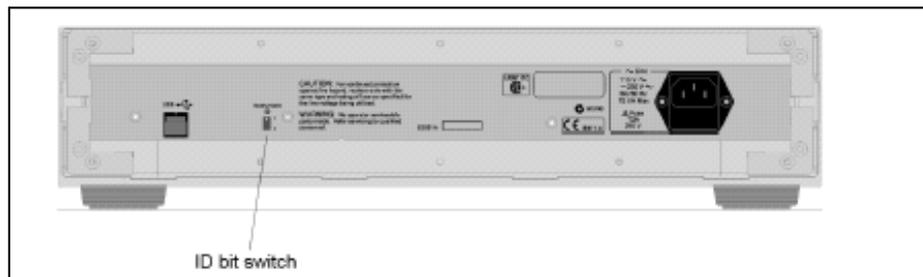
The E5091A can be connected to any one of the PNA USB ports. When first installed, Windows will automatically launch the "Add New Hardware" wizard. Click **Next** to install the E5091A test set.

Note: See the [power handling limitations](#) of the PNA USB ports.

Connect the PNA test ports to the E5091A test ports. Match PNA test port 1 to E5091A test port 1, and so forth.

Selecting ID for E5091A

The PNA can control up to two E5091A test sets. Set the Instrument ID bit switch to 1 or 2. The test sets will then be identified automatically and referred to by the DIP switch setting on the E5091A rear-panel. Change the ID bit switch setting before connecting to the PNA USB.



Power ON

Immediately after power-on, all of the port connection indicator LEDs of the E5091A go ON. Then, after the PNA detects the E5091A, the four LEDs that indicate the connected test ports remain ON. If the PNA is not powered on or if the E5091A is not connected using a USB cable, all of the LEDs stay ON.

How to make E5091A test set Control Settings

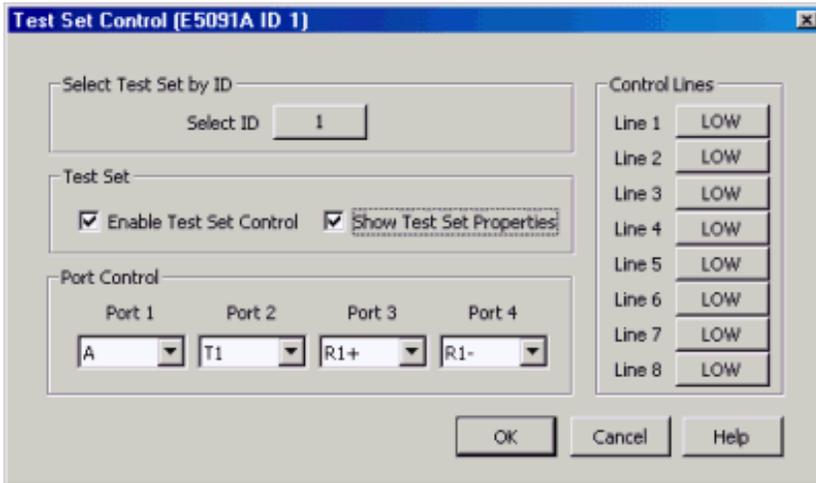
Using front-panel HARDKEY [softkey] buttons

1. Press **TRACE/CHAN**
2. then **[Channel]**
3. then **[More]**
4. then **[External test set]**
5. then **[E5091A]**

Using a mouse with PNA Menus

1. Click **Trace/Chan**
2. then **Channel**
3. then **More**
4. then **External test set**
5. then **E5091A**

No programming commands are available for this feature.



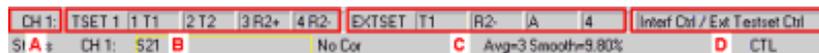
E5091A test set control dialog box help

The title of the dialog shows the test set model and ID number of the active test set..

Select ID ID of the test set to be configured. Up to two E5091A test sets can be controlled. Click to change test set ID. [Learn how to set the test set ID.](#)

Enable Test Set Control When cleared, port switching and control line settings are disabled. This selection affects all channels using the selected test set.

Show Test Set Property When checked, a second row on the status bar appears which indicates the test set that is being controlled and the current port control selection. For example, the following image shows the status bar when controlling an E5091A test set.



- Configured channel
- Port Control settings for E5091A
- Port Control settings for Z5623A K64
- Test set Label. E5091A control does not use this field. It is shared between [Interface Control](#) and [External test set Control](#). The two labels are separated by /.

Control of the second status bar is completely separate from the first status bar, which is controlled from the [View, Status Bar](#) menu.

Port Control Controls **mapping** of Physical ports to Logical ports.

- Physical ports are the port numbers that are labeled on the test set front panel. ([see N44xx test sets](#))
- Logical ports are the port numbers that are referred to by most of the PNA application prompts and dialog

boxes.

Port Mapping Notes

- Port Control and Control Line settings affect the channel of the active (selected) measurement. These settings will occur as the channel is being measured.
- Correction is NOT turned OFF when port mappings are changed. However, the **calibration is NO LONGER VALID!**

Control Lines Specifies the values of individual control lines. These general purpose control lines on the test set front-panel can be used in your test setup. Each button toggles the control line HIGH and LOW. When first opened, the selections reflect the current control lines. See your test set documentation for more information about the control lines.

OK When clicked, the changes to the dialog box are implemented and the port selections and control values are immediately sent to the specified test sets. The Port Control and Control line settings are stored with other channel data and used when those channels are swept.

Cancel (or Escape) Changes to the dialog are not implemented and revert to the settings before the dialog box was opened.

Calibrating with the E5091A

The following are a few changes in the way you calibrate the PNA with the E5091A connected:

1. Create the measurements for the channel and configure the Port Control (switching) on the E5091A Test Set Control dialog box. Enable **Show Test Set Property**.
2. To calibrate, start the Calibration wizard and select a Calibration method (ECAL, SmartCal, Unguided).
3. Select the DUT connectors that are used at the E5091A measurement reference plane.
4. When prompted to connect a standard to a PNA port, instead connect the standard to the E5091A port as indicated on the test set status bar. For example, when the cal wizard prompts to connect the standard to port 1, if the status bar indicates **1 A**, the connect the standard to port A of the E5091A.

Last Modified:

7-Apr-2009	Added 7/9 port only, and 4-port PNAs
4-Sep-2008	Removed legacy content
18-Jun-2007	MX added UI

External Multiport Test Set Control

- [Supported Test Sets](#)
 - [Option 551](#)
 - [E5091A](#) (separate topic)
- Procedure
 1. [Connect Test Set](#)
 2. [Restart as Multiport](#)
 3. [Optional External Test Set Control Settings](#)
- [External Test Set Control and other PNA Functions](#)

Other System Configuration Topics

Supported Test Sets

The list of test sets that provide integrated solutions with the PNA is constantly growing. For a current list of supported multiport test sets, see www.agilent.com/find/multiport

Option 551

- **With** Option 551 enabled on your PNA, **any supported multiport test set** (such as the U3042A E12) can be controlled directly from the PNA to make fully integrated measurements at ALL of the available test ports. To understand what test ports are available to source and receive, see the test set documentation.
- **Without** Option 551, basic operation depends on the number of PNA test ports.
 - For a 2-port PNA, configure two available test ports.
 - For a 4-port PNA, configure four available test ports.

Option 550 is no longer available.

Note: By default, the system logical test ports are mapped as follows:

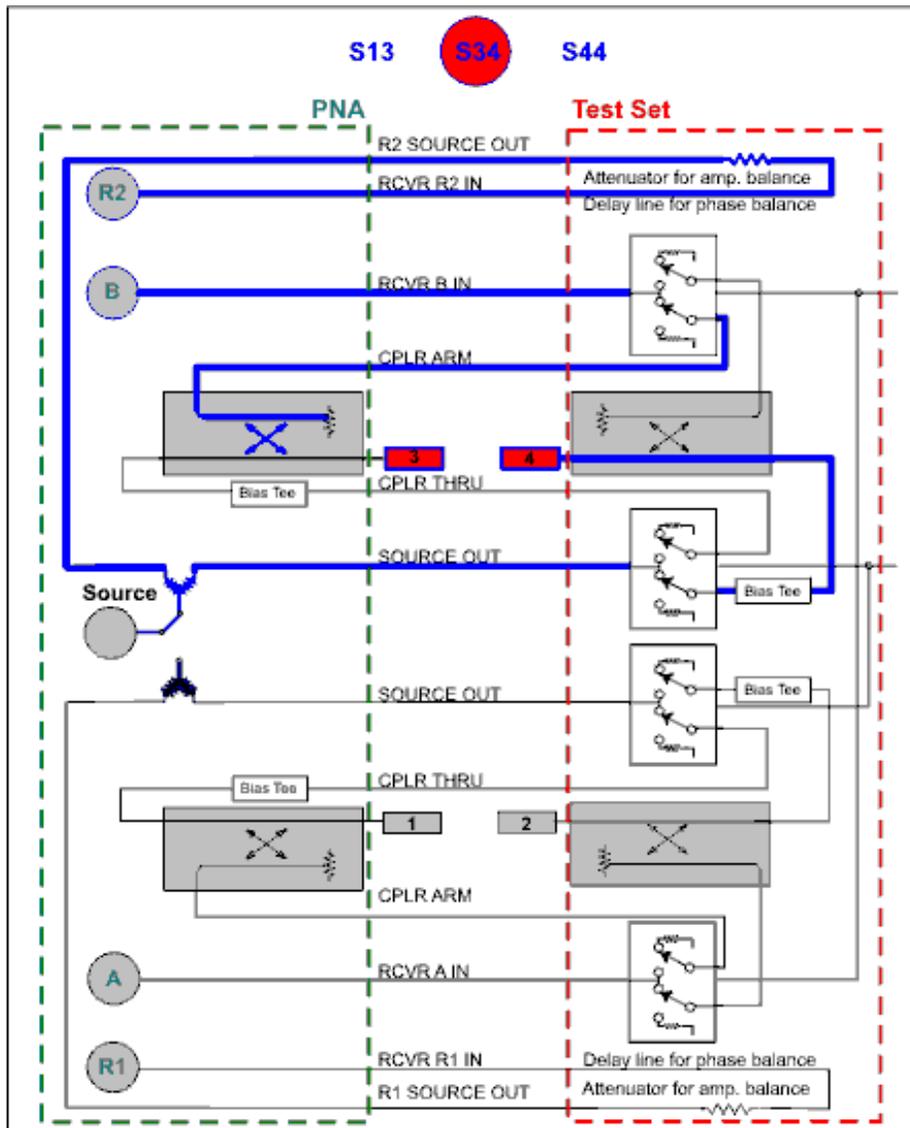


- **Port 1** - PNA port 1
- **Port 2** - Test Set port 2
- **Port 3** - PNA port 2
- **Port 4** - Test Set port 4

The ports can be **remapped** using the [Port Control Setting](#).

Block Diagram of a 2-port PNA and N44xx Test Set

Click one of the S-parameters to see switch and path changes. Because the test set does NOT contain receivers, [measurement speed](#) and [calibration](#) can be affected.



Procedure - How to enable full Multiport Capability

1. [Enable Option 551.](#)
2. Connect the test set to the PNA using the documentation that was shipped with the test set.
3. [Restart as Multiport PNA](#)
4. [Make optional External Test Set Control Settings](#)

Connect and Configure the Test Set

Connect the test set to the PNA using the test set documentation. Most test set documentation can be found at www.Agilent.com

Test Set I/O-controlled test sets

Test sets that are controlled using the [Test Set I/O connector](#), have NO return communication capability. The PNA sends commands out the rear panel connector. It is assumed that the test set is responding appropriately. The "Active" LED, located on the test set front panel, should light when the test set is addressed in Multiport Mode or manual operation. When the test set is not in use, the Active LED will be OFF.

GPIB-controlled test sets

Connect the test set to the GPIB using one of the following methods:

- If the **PNA will NOT be controlled** by a remote computer using GPIB, then the test set can be connected directly to the PNA GPIB port. The PNA is automatically switched to [System Controller](#) mode.
- If the **PNA WILL be controlled** by a remote computer using GPIB, [then learn how to connect the test set](#)

Restart as Multiport PNA

How to Enable Multiport capability

Note: If Option 551 has not been enabled, the following **Multiport Capability** menu selection will NOT be available.

Using front-panel HARDKEY [softkey] buttons

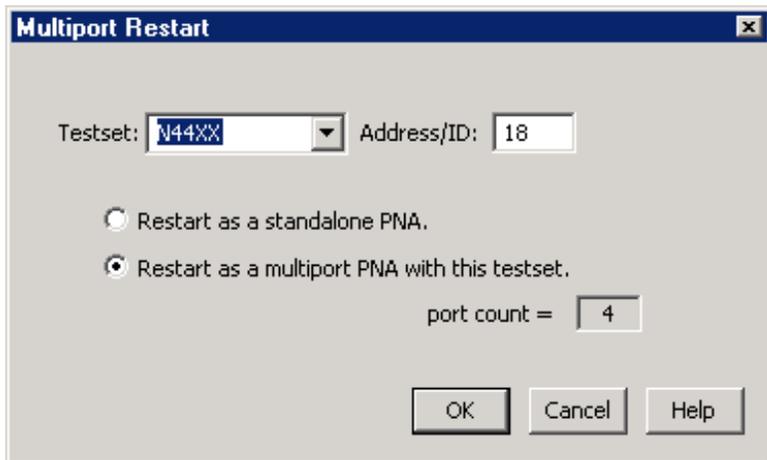
1. Press **SYSTEM**
2. then **[Configure]**
3. then **[Multiport Capability]**

Using a mouse with PNA Menus

1. Click **Utility**
2. then **System**
3. then **Configure**
4. then **Multiport Capability**

[Programming Commands](#)

[Multiport Restart dialog box help](#)



See Also [External Test Set Control and other PNA Functions](#)

After the test set is connected and [PNA Option 551 is enabled](#), the following settings are used to enable Multiport operation.

Test Set Select the test set file to load. Only the files that are appropriate for use on that PNA model are displayed.

To Add a new Test Set file:

1. On the PNA, click **File**, then **Exit** to quit the PNA application.
2. Download the Test Set file from: <http://na.tm.agilent.com/multiport/testsets/>
3. Save it to the PNA c:\users\public\network analyzer\testsets
4. Start the PNA application.
5. Click **Utility**, then **System**, then **Configure**, then **Multiport Capability**.
6. In the Multiport Restart dialog, click **Restart as multiport PNA with this testset**.
7. The new test set should now be visible from the **Testset:** menu.

Address Enter the test set address if the test set is connected to GPIB. Connections over the Test Set I/O connector are determined by their sequence.

- **Restart as a standalone PNA.** The PNA shuts down and restarts as a standard PNA. If the test set is left connected to the PNA, switch the test set OFF, then back ON to ensure that the test set routes signals to ports 1 and 2 of the PNA. In this condition, there is more loss in the test paths than without a test set connected. If the power switch is OFF, there is SIGNIFICANTLY more loss in the test paths.
- **Restart as a multiport PNA with this testset.** The PNA shuts down and restarts as a multiport PNA with the selected test set.

Click **OK** The PNA shuts down and restarts in the selected configuration.

To learn how to change port mapping, see [Port Control](#).

Problems

If the PNA cannot find the test set, the following error is displayed on the PNA:

GPIB ERROR Address xx cannot open VISA session.

To correct the problem, verify the following:

- The test set is connected to the PNA using [one of the methods described above](#).
- The correct test set address is set.
- The test set is turned ON.

Important: After the problem has been fixed:

1. On the External Test Set Control dialog, click [Enable Test Set Control](#).
2. Restart Triggering - click **Sweep, Trigger, Continuous**.
3. The PNA again tries to find the test set.

External Test Set Control Settings

The following External Test Set Control Settings are used to configure Multiport test sets. For the N44xx test sets, the only setting that is necessary is port control.

How to access the External Test Set Control Settings

Using front-panel HARDKEY [softkey] buttons

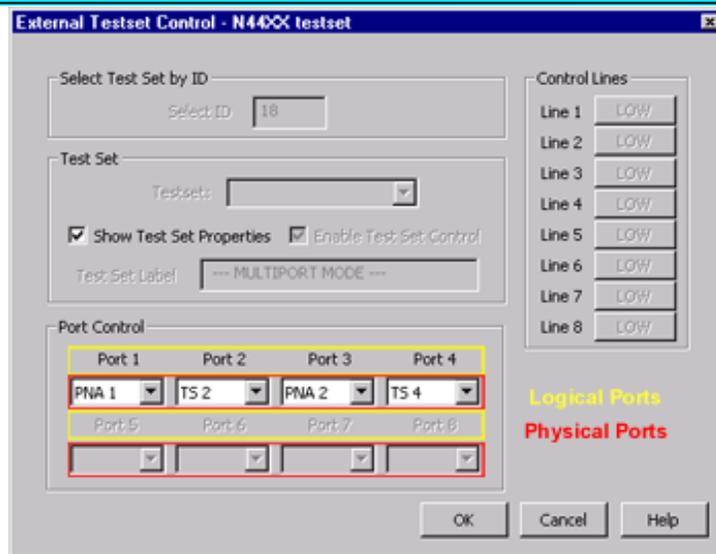
1. Press TRACE/CHAN
2. then [Channel]
3. then [Hardware Setup]
4. then [External Testset]
5. then [Other External Test Sets]

Using a mouse with PNA Menus

1. Click Trace/Chan
2. then Channel
3. then Hardware Setup
4. then External Testset
5. then Other

[Programming Commands](#)

External Test Set Control dialog box help



Important Notes:

- With Option 551, **first** use the [Multiport Restart](#) dialog to **Restart as Multiport PNA with this test set**. The test set file is loaded and the test set is enabled automatically.
- When using GPIB to control an external test set, the PNA is automatically put in [System Controller mode](#).
- See also [External Test Set Control and other PNA Functions](#)

Select ID

- For N44xx test sets: the GPIB address
- For other Multiport test sets: either GPIB address or 0 for Test Set I/O controlled test sets.

Enable Test Set Control When cleared, port switching and control line settings are disabled. This selection affects all channels using the selected test set. When checked, the 'Show Test Set Properties' checkbox is also checked automatically.

Load Test Set File For operating **without** Option 551.

If your Test Set is not visible, see [Add a new Test Set](#).

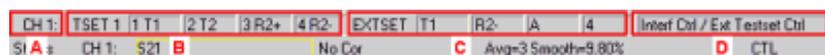
The selected test set file is loaded.

1. Navigate to the folder: C:/Program Files/Agilent/Network Analyzer/testsets/
2. Select a test set control file.

The title of the dialog shows the model of the test set file that is currently loaded.

[See a list of supported test sets.](#)

Show Test Set Properties This box becomes checked by default when the Enable Test Set Control is checked. When checked, a second row on the [status bar](#) appears which indicates the test set that is being controlled and the current port control selection. For example, the following image shows the status bar when controlling an [E5091A](#) test set and a Z5623A K64 test set:



- A. Configured channel
- B. Port Control settings for E5091A
- C. Port Control settings for Z5623A K64
- D. Test Set Label. This field is shared between [Interface Control](#) and External Test Set Control. The two labels are separated by */*.

Control of the second status bar is completely separate from the first status bar, which is controlled from the [View, Status Bar](#) menu.

Test Set Label NOT available with option 551. Add text to appear on the second status bar when **Show Test Set Properties** is checked. See image above.

Port Control Controls **mapping** of Physical ports to Logical ports. ([Refer to image of dialog box above.](#))

- Physical ports are the port numbers that are labeled on the test set front panel.
- Logical ports are the port numbers that are referred to by most of the PNA application prompts and dialog boxes.

Port Mapping Notes

- Port Control and Control Line settings effect the channel of the active (selected) measurement. These settings occur as the channel is being measured.
- Correction is turned OFF when port mappings are changed.
- After the physical ports are mapped to logical ports, all PNA references to port numbers refer to LOGICAL port numbers. The only exception to this is during calibrations.

Control Lines For use with the U30xxA test sets. Specifies the values of individual control lines. These general purpose control lines on the test set front-panel can be used in your test setup. Each button toggles the control line HIGH and LOW. When first opened, the selections reflect the current control lines. See your test set documentation for more information about the control lines.

OK When clicked, the changes to the dialog box are implemented and the port selections and control values are immediately sent to the specified test set. The Port Control and Control line settings are stored with other channel data and used when those channels are swept.

Cancel (or Escape) Changes to the dialog are not implemented and revert to the settings before the dialog box was opened.

External Test Sets and other PNA Functions

The following features may work differently with a test set connected to the PNA.

Remote Commands

- See [SCPI](#) and [COM](#) commands for controlling an External Test Set.
- Use [*OPT?](#) (SCPI) or [NumberofPorts](#) (COM) to query the number of ports for a PNA/External Test set.
- Use [logical receiver notation](#) to refer to double-digit ports.
- Use [CALC:PAR:DEF:EXT](#) instead of CALC:PAR:DEF.

Sweep Settings

To compensate for additional cable lengths:

- Set to [Stepped sweep](#)
- Set [Dwell time](#) to at least 5 microseconds.

Interface Control

When both [Interface Control](#) and External Test Set Control are configured, the commands on the Interface Control **Before Sweep Start** tab are sent out before any External Test Set Control commands are executed on that channel. Similarly, commands on the **After Sweep End** tab are sent after Test Set Control commands.

Calibration

With an External Test Set connected, calibration is performed exactly like a PNA with the following exceptions:

- Correction is turned OFF when port mappings are modified. This also applies to Source Power Cal.
- Beginning with PNA Rev. A.07.50, for [TRL Cal](#), [QSOLT](#), or [Unknown Thru](#) cals with external test sets:
 - Use of a [Delta Match Calibration](#) from a User Cal Set is NOT required. However, for PNA-L models that require Delta match, a [Global Delta Match Cal](#) must be present. The Global Delta Match Cal can only be performed in stand-alone mode.
 - You can NOT perform any of those 3 cal types on JUST a pair of ports that share a test port receiver, such as Port 1 and Port 2 of a [4-port system](#). You would need to include an additional port in the calibration.
- With an External Test Set connected, you may be required to perform more than [3 THRU connections](#).
- A test set such as the Z5623A K44 which is used with 4-port PNA models, does not terminate ports that are not currently in the source path. Because a ports load match on this system is not constant for all possible ports it can be paired with, when calibrating more than two total ports it may be necessary to make Thru measurements on more than the usual minimum number of Thru paths for a PNA calibration. The PNA will ensure that multiport calibrations use a sufficient set of Thru paths so that the calibration can correct for those variations in load match on this type of multiport system.
- As with ALL PNA calibrations, when error correction is ON, both forward and reverse sweeps are required for EACH port pair that is corrected, even if only one reflection measurement is displayed. For example, any displayed measurement with full 4-port calibration ON will require 12 measurement sweeps. [Learn more.](#)

Source Power Cal

[Source power calibration](#) involves adjusting the source so that the power at an output port is flat across a frequency range. Because of additional loss through some of the test set paths, it may NOT be possible to obtain corrected output power because of limitations on the source signal.

During a Source Power Cal, you are prompted when and where to connect the power sensor. When one of the supported test sets are connected, the prompt refers to the PHYSICAL port number, NOT the LOGICAL port number. To help with translating physical to logical port mappings, enable [Show Test Set Properties](#).

Measurements with Shared Receivers

External test sets do not contain receivers. The PNA receivers are always used to measure signals at the external test set ports. Therefore, when a channel contains two measurements that share a PNA [test port receiver](#), additional sweeps are necessary.

For example, to make S34 and S44 measurements in the same channel with correction OFF:

- On a 4-port PNA, only ONE sweep is required using the C (port 3), D (port4), and R (reference for All receivers).
- On a N44xx system, TWO sweeps are required since both measurements use the B and R2 receivers. [See](#)

[interactive block diagram above.](#)

Create Ratioed and Unratioed Measurements

When using an external test set, it IS possible to create a Ratioed measurement using two logical receivers that share the same physical PNA receiver. However, this measurement data is NOT valid. Invalid measurement traces show all data at -200 dB (in Log mag format). [Learn about Logical Receiver Notation](#)

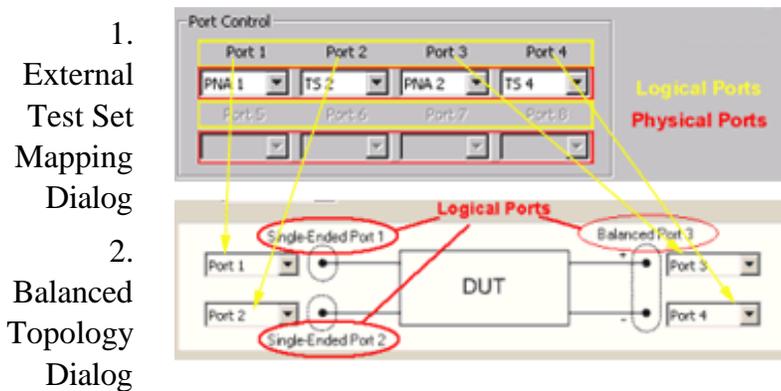
Logical Port References

When an external test set is enabled, all references to PNA port numbers and test set port numbers (except during calibrations) refer to LOGICAL port numbers. Logical ports can be remapped using the [Port Control settings](#). During a calibration, you are prompted to connect standards to physical port numbers.

Balanced Port Mapping

"Logical Ports" is a term that is used with both External Test Sets and balanced measurements. While the concept is the same, they refer to different scenarios. The two can be easily confused when making Balanced measurements with an External Test Set connected. The important principle to remember is the order in which the logical ports for each are mapped:

1. In the [External Test Set - Port Control settings dialog](#), the physical PNA ports and test set ports are mapped to logical ports as noted above.
2. In the [Balanced Topology Dialog](#), the new (step 1) logical ports are mapped again to become Balanced logical ports.



Preset

[Instrument Preset](#) will reset [Port Control](#) settings to defaults and remove the [test set label](#). All other settings remain. To maintain port control settings and the test set label, create a [User Preset](#).

Instrument State Save and Recall

[Instrument State files](#) include Test Set model, Enable and Status bar settings, and Port mappings and DUT control values for each channel.

If an Instrument State recall requires that a test set configuration file be loaded, recall time may be significant. For example, this would occur if a 2-port PNA with attached test set is configured as a 2-port PNA and then recalls a state file which requires 4-port operation.

Recall Cal Sets

If a Cal Set is saved while an external test set is enabled, when the Cal Set is recalled, then the external test set must be enabled or an error message is displayed.

Copy Channel

[Copy Channel](#) copies all relevant test set data from the source channel to the target channel.

Applications

No PNA applications are supported with External Test Set Control. These include FCA (opt 083), SMC (opt 082), GCA (opt 086), NFA (opt 029), Pulsed (opt H08).

Print

Port mapping information appears on the [Channel Settings Table](#) when printing.

Save sNp Files

To save sNp data with an [external test set](#) enabled, click File, [Save As](#), then select **Snp File (*.s*p)**, then complete the [Choose ports dialog](#).

Last modified:

25-Nov-2013	Updated for Win7
9-Apr-2012	Removed links for older models
20-Sep-2010	Added links for programming
21-Apr-2010	Updated per RD
23-Nov-2009	Edited Cals for 7.5
12-Sep-2008	Added K44 image and cal note
4-Sep-2008	Removed legacy content
15-Jan-2008	No App support
19-Nov-2007	Note to Add TS files
9-Nov-2007	Fixed S13 image and added delta match notes
17-Sep-2007	Added note for PNA-X test set files
23-Feb-2007	Modified DM Cal for 6.20.
15-Jan2007	MX Added UI
10/16/06	Added clarification to opts.
9/18/6	MQ Many edits for Opt 551
9/12/06	Added link to programming commands

Display Colors

You can modify the colors that are used to draw various elements on the PNA screen and on a hardcopy print of the display.

See Also

[Print Preview](#)

How to modify DISPLAY Colors

These settings can also be accessed from the [Preferences dialog box](#).

Using front-panel HARDKEY [softkey] buttons

1. Press **DISPLAY**
2. then **[More]**
3. then **[Display Colors]**

Using a mouse with PNA Menus

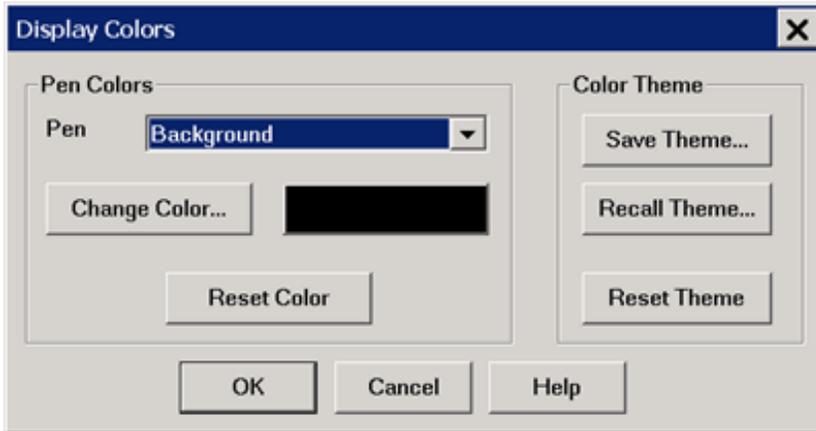
1. Click **Response**
2. then **Display**
3. then **Display Colors**

How to modify PRINT Colors

1. Press **PRINT**
2. then **[Print Colors]**

1. Click **Utility**
2. then **Print**
3. then **Print Colors**

 **Programming Commands** 



Display and Print Colors dialog box help

The Display Colors and Print Colors dialog boxes function in exactly the same manner. See [Print Preview](#) procedure below.

Pen

"Pen" is a term used to describe the various elements. Each pen can have a unique color.

You can change the color of the following pens:

- Background - The background color of the inactive windows.
- **New** Active Background - The background color of the active window.
- Grid - The inner lines of all grids in all windows, and the grid frame in inactive windows.
- Active Labels, Grid Frame - The labels and grid frame colors in the active window. **Note:** when this pen is selected, the current window becomes inactive. Therefore, changes for this pen color will not be visible until **OK** is pressed.
- Inactive Window Labels
- Failed Trace - [Limit Line](#) failed traces or failure indicators (dots) and the word Fail.
- The following pens for up to 8 Traces:
 - Data and Limits
 - Memory trace
 - Markers
 - Memory markers

About Trace Pens

'1st Trace' is NOT always Trace1 (**Tr1**). For example, the first trace in a window might be **Tr2** which is drawn with the "1st Trace" pen.

The first 8 traces are drawn with the defined pen colors. The next eight traces reuse the same colors, and so forth. For example, if all traces are numbered sequentially, the 9th and 17th traces are drawn using the same color as the 1st trace.

Change Color Click the button or the color swatch to launch the [Change Color](#) dialog.

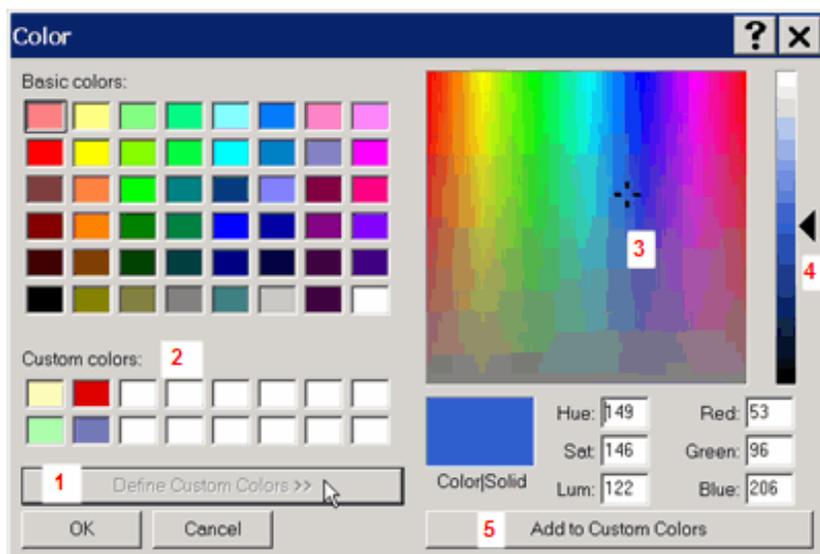
Reset Color Restores the default color for the selected pen.

Color Themes

A theme is a complete set of pens and their colors. The current theme persists until you change it. Themes can also be saved to a file and then later recalled.

- **Save Theme** Click to save the current set of pens to a file.
- **Recall Theme** Click to recall and use a saved theme.
- **Reset Theme** Click to recall the default PNA color theme.

The colors for the following Display elements can NOT be changed: toolbars, softkeys, menus, dialogs and popup messages.



Change Color dialog box help

To use a basic color, click the color from the 'Basic colors' palette, then click **OK**.

To define and use a custom color:

1. Click **Define Custom Colors>>** to open the right side of the dialog.
2. Optionally, pick a Custom color slot to replace. Otherwise, the replacement will occur at the first slot location and continue with subsequent custom color definitions.
3. Click the color pane, or drag the crosshairs, to the location of the custom color.
4. Drag the arrow to the desired saturation level of the custom color.
5. Click **Add to Custom Colors**
6. Continue to define more colors, or click **OK** to close the Color dialog.

After a custom color has been assigned to a PNA pen, the custom color can be changed. The PNA pen color remains unchanged.

Print Preview Procedure

Use the following procedure to preview your Print Colors on the PNA screen:

1. From the Print Colors dialog, select **Reset Theme** then **Save Theme**. Name the new theme "MyPrintTheme.colors". This will give you a starting point equal to the default print colors.
2. Launch the Display Colors dialog, select **Recall Theme**, then select "MyPrintTheme.colors". The display will now show the default print theme.
3. Customize the display colors. You will be previewing how the hardcopy will appear when printed.
4. Save the customized display colors to "MyPrintTheme.colors".
5. Go to the Print Colors dialog and Recall "MyPrintTheme.colors".

Last Modified:

22-Feb-2010 Added Active background (A.09.20)

7-Aug-2009 MX New topic

Mechanical Devices

- [Overview](#)
- [How to access Mechanical Devices settings](#)
- [Mechanical Devices dialog](#)

Other System Configuration Topics

Overview

Note: To prevent premature wear, the PNA does not allow attenuators or other mechanical switches to switch continuously.

These mechanical devices are set for the entire channel. When more than one channel is used, and a mechanical device setting is NOT the same for all channels, only the ACTIVE channel is allowed to sweep. All other channels are **Blocked** - NOT allowed to sweep. Blocked channels will resume sweeping when they are made ACTIVE, or when the conflict is resolved.

Press **TRIGGER** then [**Restart**] to cause ALL channels to sweep once. Then the active channel will resume sweeping continuously.

Before PNA Rev. A.09.20, conflicting channels were put in trigger Hold.

The Mechanical Devices dialog shows the settings of all of the switches and attenuators in the PNA. The settings for all active channels are shown side-by-side for easy comparison. This dialog allows you to determine the settings which would cause mechanical devices to switch between states on consecutive sweeps, potentially leading to device wear-out. It also allows you to determine if the conflict can be resolved to enable continuous sweeps on all channels.

The following are the mechanical devices that are potentially shown in the dialog. These components may not appear in your PNA model:

- Port 1 through Port 4 Bypass Switches
- Port 1 through Port 4 Source Attenuator settings
- Receiver A through Receiver R Attenuator settings
- Port 1 Noise Tuner Switch and Port 2 Noise Receiver Switch

How to access Mechanical Devices settings

Using front-panel HARDKEY [softkey] buttons

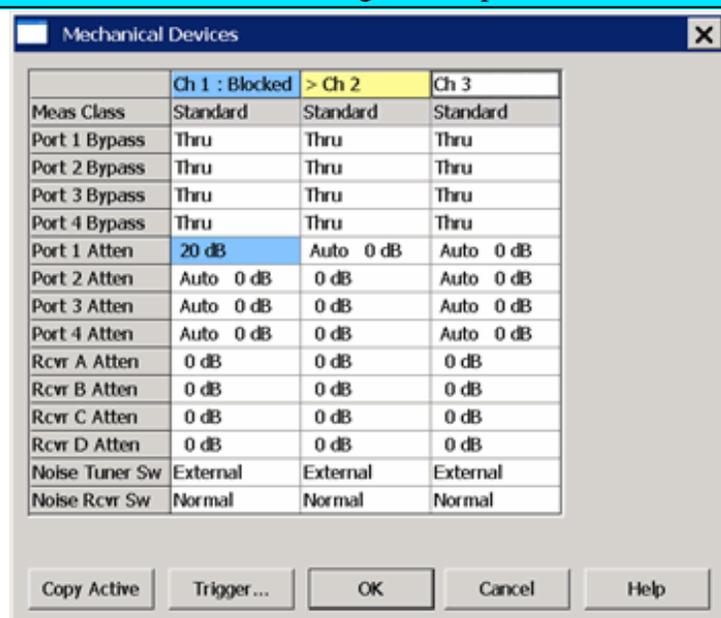
1. Press CHAN
2. then [Channel]
3. then [Hardware Setup]
4. then [Mechanical Devices]

Using a mouse with PNA Menus

1. Click Trace/Chan
2. then Channel
3. then Hardware Setup
4. then Mechanical Devices

No programming commands are available for this feature.

Mechanical Devices dialog box help



[See Mechanical Devices Overview \(scroll up\)](#)

The devices that appear in the table depend on the PNA model and options.

> **Yellow** highlighted cell indicates the Active channel.

Blue highlighted cells indicate the following:

- The channel is NOT able to sweep. **Blocked** is shown in the top row.
- The highlighted device settings differ from that of the sweeping channels.

To modify entries in the table, click a cell.

When a selection is changed, the new setting is applied immediately.

If [Port Power is coupled](#), a dialog prompts if coupling should be turned OFF.

Not supported

- [Measurement Class](#) can NOT be changed from this dialog.
- The dialog does NOT report device settings for [multiport test sets](#).
- This dialog does NOT report device settings for [external sources](#).

Copy Active Available ONLY when there is a conflict which causes at least one channel to be Blocked. When clicked, the mechanical device settings of the **Active** channel are copied to the Blocked channels. A warning message appears to remind you that power to one or more channels may be increased. **Exception:** When one or more Noise channels are present, then the settings of the two Noise switches are determined by the lowest-numbered Noise channel if none are the active channel.

Trigger Launches the Trigger dialog box.

OK Closes the dialog box.

Cancel Does not apply changes that were made, and closes the dialog box.

Last Modified:

11-Nov-2013	Added copy active
23-Feb-2010	Added Channel Blocking (A.09.20)
31-Jul-2009	MX New topic

Power Limit and Power Offset

- [Overview](#)
- [How to access Power Limit and Power Offset settings](#)

Other System Topics

Overview

Power Limit (Global scope)

Global power limit sets a maximum source power level for individual PNA ports. This value limits port power for all channels and all applications. Power levels that attempt to exceed the power limit is clipped at the limit.

Notes

- The power limit can NOT be set for power levels which are below the power level that is required by the PNA to achieve phase lock - approximately -30 dBm.
- Because [Fast Sweep mode](#) allows power spiking, it is NOT allowed when a power limit is enabled.
- Components that are added to the RF path are accounted for by entering their loss (negative) or gain (positive) in the [Power Offset](#) section of the dialog box.
- [PNA Applications](#) may change RF path components. For example, IMD for Converters may change the combiner path and add an amplifier for LO input. Compensation is NOT made for these changes and port power may exceed the power limit or port power may be clipped unnecessarily.
- Power limiting does NOT clip power spikes that may occur during [frequency band crossings](#).
- [External test set ports](#) are also included for power limiting.

Power Offset (Channel scope)

Power Offset provides a method of compensating port power for added attenuation or amplification in the source path. The result is that power at the specified port, all dialogs, and annotation, reflects the added components.

How to access the Offsets and Limits settings

Also accessed through the [PNA Preferences](#) dialog.

Using front-panel HARDKEY [softkey] buttons

1. Press **POWER**
2. then **[Power and Attenuators]**
3. then **[Offsets and Limits]**

Using a mouse with PNA Menus

1. Click **Stimulus**
2. then **Power**
3. then **Power and Attenuators**
4. then **Offsets and Limits**

Programming Commands

	Power Limit		Power Offset: Channel 1			
	State	Limit	Source Power	Power Offset	Port Power	Source Cal
Port 1	On	0.00 dBm	-15.00 dBm	3.00 dB	-12.00 dBm	Off
Port 2	Off	100.00 dBm	-15.00 dBm	0.00 dB	-15.00 dBm	Off
Port 3	Off	100.00 dBm	-15.00 dBm	0.00 dB	-15.00 dBm	Off
Port 4	Off	100.00 dBm	-15.00 dBm	0.00 dB	-15.00 dBm	Off
Port 1 Src2	Off	100.00 dBm	-15.00 dBm	0.00 dB	-15.00 dBm	Off

Offsets and Limits dialog box help

Click a **WHITE** cell to change values. **Shaded cells** can **NOT** be changed.

[Remote commands](#) can be sent to lock and unlock the dialog box (UI) settings.

Power Limit

Limits the source power at each PNA port for **ALL** channels. Use this feature to protect DUTs that are sensitive to overpowering at the input. Power levels that exceed the limit at the specified port are clipped at the limit and an error message is displayed on the screen.

The Power Limit settings survive [Instrument Preset](#). When an Instrument State is [recalled](#), the current Power Limit settings are applied to the recalled state.

To learn more, see [Power Limit Overview](#) (scroll up).

State / Limit

- **ON** - Power is limited to the adjacent value at the specified source port.

- **OFF** - Power is NOT limited to this value, but to the maximum power of the PNA source.

For PNA models with a second internal source, the **Port 1 Src2** Power Limit setting is NEVER available. Make the setting at the standard **Port 1**.

Power Offset

Power Offset provides a method of compensating port power for added attenuation or amplification in the source path. The result is that power at the specified port, all dialogs, and annotation reflects the added components.

- For amplification, use positive offset.
- For attenuation, use negative offset.

Important Note: Power Offset is added AUTOMATICALLY when a [Source Power Calibration](#), [Guided Power Cal](#), or [Power Compensation](#) is ON with Fixture Embed/Deembed. If you are NOT seeing the correct power level at your DUT, view the power Offset column in this dialog for unexpected offsets.

Optionally change the Source Power or Port Power values so that the following equation reflects your requirement:

$$\text{Source Power} + \text{Power Offset} = \text{Port Power}$$

Source Cal ON / OFF

Notes

- Power Offset can be used with [Power Sweeps](#). When a power sweep is enabled, the Start and Stop power levels are reported in this dialog.
- When port power offsets are used, port powers are automatically [uncoupled](#). Port powers may not be coupled again until all port offsets are zero.

OK Closes the dialog box.

Last Modified:

15-Nov-2011 Added Power cal notes
31-Jul-2009 MX New topic

Setting System Impedance

The system impedance can be changed for measuring devices with an impedance other than 50 ohms, such as waveguide devices. The PNA mathematically transforms and displays the measurement data as though the PNA ports were the specified impedance value. Physically, the test ports are always about 50 ohms.

How to change the System Impedance

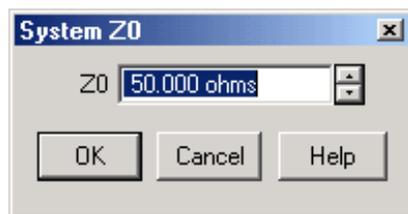
Using front-panel HARDKEY [softkey] buttons

1. Press **SYSTEM**
2. then [**Configure**]
3. then [**System Z0**]

Using a mouse with PNA Menus

1. Click **Utility**
2. then **System**
3. then **Configure**
4. then **System Z0**

Programming Commands



System Z0 dialog box help

Allows you to change the system impedance (default setting is 50 ohms).

Z0 Displays the current system impedance.

For 75 ohm devices:

1. Change the system Z0 to 75 ohms.
2. Connect minimum loss pads (75 ohm impedance) between the analyzer and the DUT to minimize the physical mismatch.
3. Perform a calibration with 75 ohm calibration standards.

For waveguide devices

Beginning with A.09.50, when selecting a Cal Kit with an impedance other than 50 ohms

(Waveguide = 1 ohm), it is **NO LONGER NECESSARY** to change the [System Impedance](#) setting before performing a calibration. The impedance for the calibration is now derived from the Cal Kit '[Connector](#)' impedance setting.

Last modified:

10-Dec-2013	Added connector to Impedance setting
11-Sep-2012	Add Z0 note (BH)
14-Jul-2011	Edit WG restriction
3-Sep-2008	Removed legacy content
9/27/06	MX Added UI
9/12/06	Added link to programming commands

PNA Application Notes

The following links require an **Internet connection**.

Note: Check out the multimedia PNA Demo presentations, including ['Network Analyzer Basics'](#).

Calibrations

AN1287-11 [Specifying Calibration Standards and Kits for Agilent Vector Network Analyzers \(5989-4840EN\)](#)

PN8510-8A [TRL Calibration for Non-Coaxial Measurements \(5091-3645E\)](#)

[Calibrating Standards for In-Fixture Device Characterization \(White Paper\) \(5989-3245EN\)](#)

[Electronic vs. Mechanical Calibration kits: Calibration methods and accuracy \(White Paper\) \(5988-9477EN\)](#)

[On-Wafer Calibration Using a 4-port, 20 GHz PNA-L Network Analyzer \(N5230A Option 240/245\) \(5989-2287EN\)](#)

ECal

[Agilent Electronic vs. Mechanical Calibration Kits: Calibration Methods and Accuracy \(5988-9477EN\)](#)

[User Characterization: Electronic Calibration Feature Allows Users to Customize to Specific Needs \(5988-9478EN\)](#)

Embedding / De-embedding

[De-embedding and Embedding S-Parameter Networks Using a Vector Network Analyzer \(5980-2784EN\)](#)

Amplifier Measurements

AN1408-7 [Amplifier Linear and Gain Measurements \(5988-8644EN\)](#)

AN1408-8 [Amplifier Swept-Harmonic Measurements \(5988-9473EN\)](#)

AN1408-9 [Amplifier and CW Swept Intermodulation-Distortion Measurements \(5988-9474EN\)](#)

AN1408-10 [High-power measurements using the PNA \(5989-1349EN\)](#)

AN1408-16 [Power-Added Efficiency \(PAE\) 5989-7293EN](#)

AN1408-17 [Making Accurate IMD Measurements with the PNA-X Network Analyzer \(5989-7265EN\)](#)

AN1408-19 [High Power Amplifier Measurements Using NVNA](#)

Antenna Measurements

[Triggering PNA Microwave Network Analyzers for Antenna Measurements \(5988-9518EN\)](#)

[New Network Analyzer Methodologies in Antenna/RCS Measurements \(5989-1937EN\)](#)

[Pulsed Antenna Measurements Using PNA Network Analyzers \(5989-0221EN\)](#)

[Antenna and RCS Configurations \(White Paper\) \(5989-0220EN\)](#)

[Radar Measurements \(Application Note\) \(5989-7575EN\)](#)

Balanced Measurements (Although the following refer to the ENA, they are also relevant to the PNA.)

[On-wafer Balanced Component Measurement with the Cascade Microtech Probing System \(5988-5886EN\)](#)

[Network De-embedding/Embedding and Balanced Measurement \(5988-4923EN\)](#)

[Backplane Differential Channel Microprobe Characterization in Time and Frequency Domains \(White Paper\) \(5989-3248EN\)](#)

Mixer Measurements

AN1408-1 [Mixer Transmission Measurements Using the Frequency Conversion Application \(5988-8642EN\)](#)

AN1408-2 [Mixer Conversion-Loss and Group Delay Measurement Techniques and Comparisons \(5988-9619EN\)](#)

AN1408-3 [Improving Measurement and Calibration Accuracy Using the Frequency Converter Application \(5988-9642EN\)](#)

AN1408-18 [Measuring Group Delay of Frequency Converters with Embedded Local Oscillators \(5989-7385EN\)](#)

[Comparison of Mixer Characterization using New Vector Characterization Techniques \(5988-7827EN\)](#)

[Novel Method for Vector Mixer Characterization and Mixer Test System Vector Error Correction \(5988-7826EN\)](#)

[Measuring Absolute Group Delay of Multistage Converters Using PNA Microwave Network Analyzers \(5989-0219EN\)](#)

Pulsed Measurements

AN1408-11 [Accurate Pulsed Measurements \(5989-0563EN\)](#)

AN1408-12 [Pulsed-RF S-Parameter Measurements Using Wideband and Narrowband Detection](#)

AN1408-21 [Active-Device Characterization in Pulsed Operation Using the PNA-x \(5990-7781EN\)](#)

[Pulsed Antenna Measurements Using PNA Network Analyzers \(5989-0221EN\)](#)

Materials Measurements

[Basics of Measuring the Dielectric Properties of Materials \(5989-2589EN\)](#)

[Split Post Dielectric Resonators for Dielectric Measurements of Substrates \(5989-5384EN\)](#)

Other Measurements

AN1287-12 [Time Domain Analysis Using a Network Analyzer \(5989-5723EN\)](#)

AN1408-14 [Using the PNA Series to Analyze Lightwave Components \(5989-3385EN\)](#)

AN1408-15 [Using the PNA for Banded Millimeter-Wave Measurements \(5989-4098EN\)](#)

AN1408-19 [High Power Amplifier Measurements Using NVNA \(5990-5039EN\)](#)

AN1408-20 [High-Accuracy Noise Figure Measurements Using the PNA-X](#)

[MM-Wave Network Analyzers: Analysis of Cable Length on VNA System Performance \(5989-1941EN\)](#)

[Ultra-Low Impedance Measurements Using 2-Port Measurements \(White Paper\) \(5989-5935EN\)](#)

Modeling

[Utilizing TDR and VNA Data to Develop 4-port Frequency Dependent Models \(White Paper\) \(5989-0638EN\)](#)

[Advanced Measurements and Modeling of Differential Devices \(White Paper\) \(5989-4518EN\)](#)

Automation

AN 1408-13 [Introduction to Application Development using the PNA \(5980-2666EN\)](#)

[Connectivity Advances for Component Manufacturers \(5980-2782EN\)](#)

[The 'Need for Speed' in Component Manufacturing Test \(5980-2783EN\)](#)

4-Dec-2012	Fixed TD AN
10-Oct-2012	Added AN numbers
4-May-2012	Added PAE
20-May-2011	Added new pulse
7-Mar-2011	Added new app notes
22-Oct-2010	Added Noise Figure
14-Apr-2008	Added IMD with PNA-X.

Network Analyzer Basics

This self-paced two hour video discusses the basic concepts of Network Analysis.

The files are installed and should work on older PNA models. If the PNA link does not work, then use the internet link, which requires an internet connection.

- **From the PNA:** [Proceed with Network Analyzer Basics](#).
 - **From the Internet:** <http://wireless.agilent.com/networkanalyzers/pnademo.htm> in both streaming and downloadable format.
-

Last modified:

10/18/06 Added link to pnamemo.

Connector Care

Proper connector care is critical for accurate and repeatable measurements. The following information will help you preserve the precision and extend the life of your connectors - saving both time and money.

- [Connector Care Quick Reference Guide](#)
- [Connector Cleaning Supplies](#)
- [Safety Reminders](#)
- [About Connectors](#)
- [Gaging Fundamentals](#)
- [Connector Care Procedures](#)

See also mmWave Connector Care at http://na.tm.agilent.com/pna/connectorcare/Connector_Care.htm

Important Note

All of the copper N5247A front-panel loops are **EXTREMELY FRAGILE**.



A maximum **8 inch/lbs of torque** should be applied to these connectors.

If more torque is applied, damage to the copper semi-rigid cables **WILL** occur, causing measurement inaccuracy.

Preventing Test Port Connector Damage

Handling and Storing Connectors

Do

Keep connectors clean

Protect connectors with plastic end caps

Keep connector temperature same as analyzer

Do Not

Touch mating-plane surfaces

Set connectors contact-end down

Store connectors loose in box or drawer

Visual Inspection

Do

Inspect connectors with magnifying glass.

Look for metal debris, deep scratches or dents

Do Not

Use a connector with a bent or broken center conductor

Use a connector with deformed threads

Cleaning Connectors

Do

Clean surfaces first with clean, dry compressed air

Use lint-free swab or brush

Use minimum amount of alcohol

Clean outer conductor mating surface and threads

Do Not

Use high pressure air (>60 psi)

Use any abrasives

Allow alcohol into connector support beads

Apply lateral force to center conductor

Gaging Connectors

Do

Inspect and clean gage, gage master and device tested

Use correct torque wrench

zero gage before use

Use multiple measurements and keep record of readings

Do Not

Use an out of specification connector

Hold connector gage by the dial

Making Connections

Do

Align connectors first

Rotate only the connector nut

Use correct torque wrench

Do Not

Cross thread the connection

Twist connector body to make connection

Mate different connector types

Connector Care and Cleaning Supplies

Description	Web Site
Swabs	http://www.berkshire.com/swabs.shtml
Lint Free Cloths- Air dusters	http://www.ccrwebstore.com
Isopropyl	http://www.techspray.com
Nitrilite Gloves and Finger Cots	http://www.techni-tool.com

Safety Reminders

When cleaning connectors:

- Always use protective eyewear when using compressed air or nitrogen.
- Keep isopropyl alcohol away from heat, sparks and flame. Use with adequate ventilation. Avoid contact with eyes, skin and clothing.
- Avoid electrostatic discharge (ESD). Wear a grounded wrist strap (having a 1 M Ω series resistor) when cleaning device, cable or test port connectors.
- Cleaning connectors with alcohol shall only be done with the instruments power cord removed, and in a well-ventilated area. Allow all residual alcohol moisture to evaporate, and the fumes to dissipate prior to energizing the instrument.

About Connectors

- [Connector Service Life](#)
- [Connector Grades and Performance](#)
- [Adapters as Connector Savers](#)
- [Connector Mating Plane Surfaces](#)

Connector Service Life

Even though calibration standards, cables, and test set connectors are designed and manufactured to the highest standards, all connectors have a limited service life. This means that connectors can become defective due to wear during normal use. For best results, all connectors should be inspected and maintained to maximize their service life.

Visual Inspection should be performed each time a connection is made. Metal particles from connector threads often find their way onto the mating surface when a connection is made or disconnected. See [Inspection](#) procedure.

Cleaning the dirt and contamination from the connector mating plane surfaces and threads can extend the service life of the connector and improve the quality of your calibration and measurements. See

[Cleaning](#) procedure.

Gaging connectors not only provides assurance of proper mechanical tolerances, and thus connector performance, but also indicate situations where the potential for damage to another connector may exist. See [Gaging](#) procedure.

Proper connector care and connection techniques yield:

- Longer Service Life
- Higher Performance
- Better Repeatability

Connector Grades and Performance

The three connector grades (levels of quality) for the popular connector families are listed below. Some specialized types may not have all three grades.

- **Production** grade connectors are the lowest grade and the least expensive. It is the connector grade most commonly used on the typical device under test (DUT). It has the lowest performance of all connectors due to its loose tolerances. This means that production grade connectors should always be carefully inspected before making a connection to the analyzer. Some production grade connectors are not intended to mate with metrology grade connectors.
- **Instrument** grade is the middle grade of connectors. It is mainly used in and with test instruments, most cables and adapters, and some calibration standards. It provides long life with good performance and tighter tolerances. It may have a dielectric supported interface and therefore may not exhibit the excellent match of a metrology grade connector.
- **Metrology** grade connectors have the highest performance and the highest cost of all connector grades. This grade is used on calibration standards, verification standards, and precision adapters. Because it is a high precision connector, it can withstand many connections and disconnections and, thus, has the longest life of all connector grades. This connector grade has the closest material and geometric specifications. Pin diameter and pin depth are very closely specified. Metrology grade uses an air dielectric interface and a slotless female contact which provide the highest performance and traceability.

Note: In general, Metrology grade connectors should not be mated with Production grade connectors.

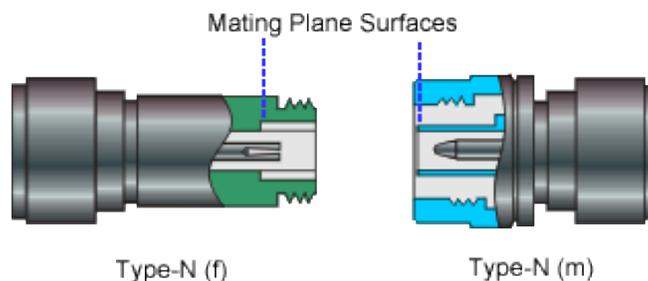
Adapters as Connector Savers

Make sure to use a high quality (Instrument grade or better) adapter when adapting a different connector type to the analyzer test ports. It is a good idea to use an adapter even when the device under test is the same connector type as the analyzer test ports. In both cases, it will help extend service life, and protect the test ports from damage and costly repair.

The adapter must be fully inspected before connecting it to the analyzer test port and inspected and cleaned frequently thereafter. Because calibration standards are connected to the adapter, the adapter should be the highest quality to provide acceptable RF performance and minimize the effects of mismatch.

Connector Mating Plane Surfaces

An important concept in RF and microwave measurements is the reference plane. For a network analyzer, this is the surface that all measurements are referenced to. At calibration, the reference plane is defined as the plane where the mating plane surfaces of the measurement port and the calibration standards meet. Good connections (and calibrations) depend on perfectly flat contact between connectors at all points on the mating plane surfaces (as shown in the following graphic).

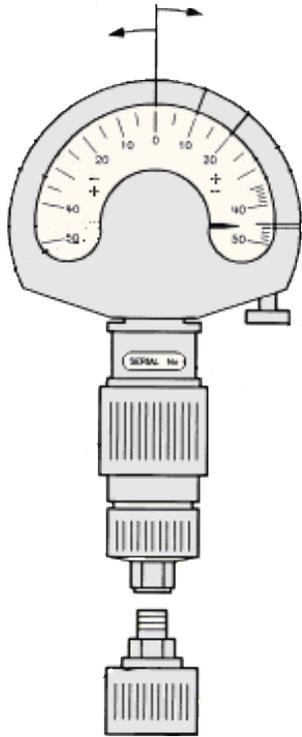


Gaging Fundamentals

Connector gages are important tools used to measure center conductor pin depth in connectors. Connector pin depth, measured in terms of recession or protrusion, is generally the distance between the mating plane and the end of the center conductor, or the shoulder of the center conductor for a stepped male pin.

Typical Connector Gage

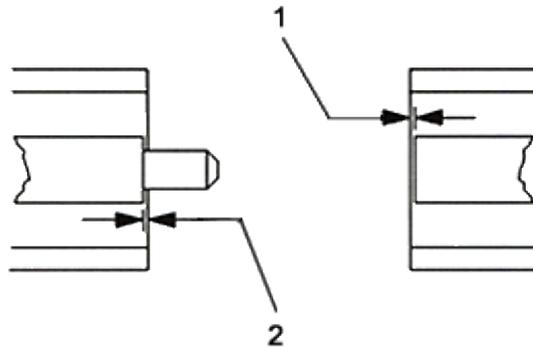
RECESSION PROTRUSION



Recession and Protrusion

Pin depth is negative (recession) if the center conductor is recessed below the outer conductor mating plane, usually referred to as the "reference plane". Pin depth is positive (protrusion) if the center conductor projects forward from the connector reference plane.

Pin Depth



1. Recession of female contact
2. Recession of male pin shoulder

Difference with Type-N Connectors

Type-N connectors have the mating plane of the center conductors offset from the connector reference plane. In this case the zero setting "gage masters" generally offset the nominal distance between the center conductor mating plane and the connector reference plane.

When to Gage Connectors

- Before using a connector or adapter the first time.
- When visual inspection or electrical performance suggests the connector interface may be out of range.
- After every 100 connections, depending on use.

Connector Gage Accuracy

Connector gages (those included with calibration and verification kits), are capable of performing coarse measurements only. This is due to the repeatability uncertainties associated with the measurement. It is important to recognize that test port connectors and calibration standards have mechanical specifications that are extremely precise. Only special gaging processes and electrical testing (performed in a calibration lab) can accurately verify the mechanical characteristics of these devices. The pin depth specifications in the Agilent calibration kit manuals provide a compromise between the pin depth accuracy required, and the accuracy of the gages. The gages shipped with calibration and verification kits allow you to measure connector pin depth and avoid damage from out-of-specification connectors.

Note: Before gaging any connector, the mechanical specifications provided with that connector or device should be checked.

To Gage Connectors

1. Wear a grounded wrist strap (having a 1 M Ω series resistor).
2. Select proper gage for device under test (DUT).
3. Inspect and clean gage, gage master, and DUT.
4. Zero the connector gage.
 - a. While holding gage by the barrel, carefully connect gage master to gage. Finger-tighten connector nut only.
 - b. Use proper torque wrench to make final connection. If needed, use additional wrench to prevent gage master (body) from turning. Gently tap the barrel to settle the gage.
 - c. The gage pointer should line up exactly with the zero mark on gage. If not, adjust "zero set" knob until gage pointer reads zero. On gages having a dial lock screw and a movable dial, loosen the dial lock screw and move the dial until the gage pointer reads zero. Gages should be zeroed before each set of measurements to make sure zero setting has not changed.

- d. Remove gage master.
5. Gage the device under test.
 - a. While holding gage by the barrel, carefully connect DUT to gage. Finger-tighten connector nut only.
 - b. Use proper torque wrench to make final connection and, if needed, use additional wrench to prevent DUT (body) from turning. Gently tap the barrel to settle the gage.
 - c. Read gage indicator dial for recession or protrusion and compare reading with device specifications.

Caution: If the gage indicates excessive protrusion or recession, the connector should be marked for disposal or sent out for repair.

6. For maximum accuracy, measure the device a minimum of three times and take an average of the readings. After each measurement, rotate the gage a quarter-turn to reduce measurement variations.
7. If there is doubt about measurement accuracy, be sure the temperatures of the parts have stabilized. Then perform the cleaning, zeroing, and measuring procedure again.

Connector Care Procedures

- [Inspecting Connectors](#)
- [Cleaning Connectors](#)
- [Making Connections](#)
- [Using a Torque Wrench](#)
- [Handling and Storing Connectors](#)

To Inspect Connectors

Wear a grounded wrist strap (having a 1 M Ω series resistor).

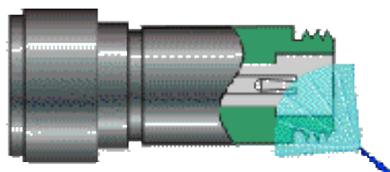
Use a magnifying glass ($\geq 10X$) and inspect connector for the following:

- Badly worn plating or deep scratches
- Deformed threads
- Metal particles on threads and mating plane surfaces
- Bent, broken, or mis-aligned center conductors
- Poor connector nut rotation

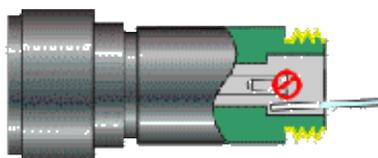
Caution: A damaged or out-of-specification device can destroy a good connector attached to it even on the first connection. Any connector with an obvious defect should be marked for disposal or sent out for repair.

To Clean Connectors

1. Wear a grounded wrist strap (having a 1 M Ω series resistor).
2. Use clean, low-pressure air to remove loose particles from mating plane surfaces and threads. Inspect connector thoroughly. If additional cleaning is required, continue with the following steps.



3. Moisten—do not saturate—a lint-free swab with isopropyl alcohol. See [Cleaning Supplies](#) for recommended type.
4. Clean contamination and debris from mating plane surfaces and threads. When cleaning interior surfaces, avoid exerting pressure on center conductor and keep swab fibers from getting trapped in the female center conductor.



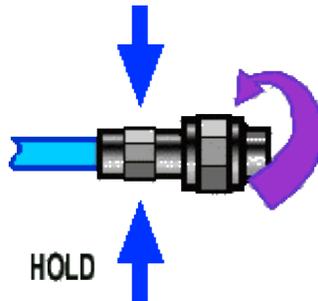
5. Let alcohol evaporate—then use compressed air to blow surfaces clean.
6. Inspect connector. Make sure no particles or residue remains.
7. If defects are still visible after cleaning, the connector itself may be damaged and should not be used. Determine the cause of damage before making further connections.

To Make Connections

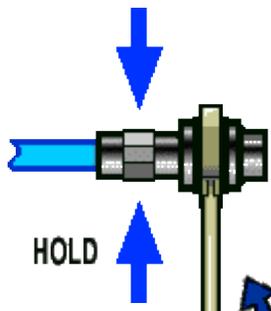
1. Wear a grounded wrist strap (having a 1 M Ω series resistor).
2. Inspect, clean, and gage connectors. All connectors must be undamaged, clean, and within mechanical specification.
3. Carefully align center axis of both devices. The center conductor pin—from the male connector—must slip concentrically into the contact finger of the female connector.



4. Carefully push the connectors straight together so they can engage smoothly. Rotate the connector nut (not the device itself) until finger-tight, being careful not to cross the threads.



5. Use a torque wrench to make final connection. Tighten until the "break" point of the torque wrench is reached. Do **not** push beyond initial break point. Use additional wrench, if needed, to prevent device body from turning.



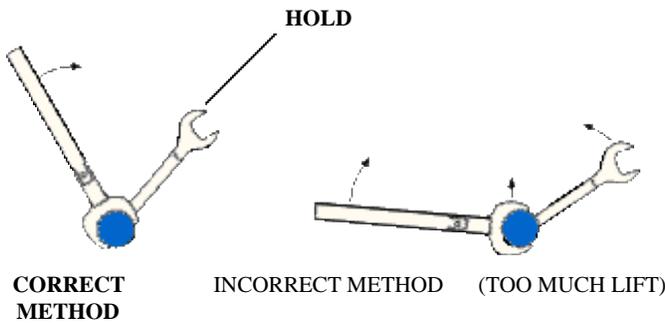
To Separate a Connection

1. Support the devices to avoid any twisting, rocking or bending force on either connector.
2. Use an open-end wrench to prevent the device body from turning.
3. Use another open-end wrench to loosen the connector nut.
4. Complete the disconnection by hand, turning only the connector nut.
5. Pull the connectors straight apart.

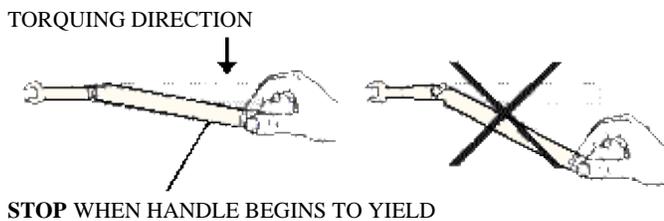
To Use a Torque Wrench

1. Make sure torque wrench is set to the correct torque setting.

2. Position torque wrench and a second wrench (to hold device or cable) within 90° of each other before applying force. Make sure to support the devices to avoid putting stress on the connectors.



3. Hold torque wrench lightly at the end of handle—then apply force perpendicular to the torque wrench handle. Tighten until the "break" point of the torque wrench is reached. Do **not** push beyond initial break point.



To Handle and Store Connectors

- Install protective end caps when connectors are not in use.
- Never store connectors, airlines, or calibration standards loose in a box. This is a common cause of connector damage.
- Keep connector temperature the same as analyzer. Holding the connector in your hand or cleaning connector with compressed air can significantly change the temperature. Wait for connector temperature to stabilize before using in calibration or measurements.
- Do not touch mating plane surfaces. Natural skin oils and microscopic particles of dirt are difficult to remove from these surfaces.
- Do not set connectors contact-end down on a hard surface. The plating and mating plane surfaces can be damaged if the interface comes in contact with any hard surface.
- Wear a grounded wrist strap and work on a grounded, conductive table mat. This helps protect the analyzer and devices from electrostatic discharge (ESD).

Electrostatic Discharge (ESD) Protection

Protection against electrostatic discharge (ESD) is essential while removing or connecting cables to the network analyzer. Static electricity can build up on your body and can easily damage sensitive internal circuit elements when discharged. Static discharges too small to be felt can cause permanent damage. To prevent damage to the instrument:

- **Always** have a grounded, conductive table mat in front of your test equipment.
- **Always** wear a grounded wrist strap, connected to a grounded conductive table mat, having a 1 M Ω resistor in series with it, when making test setup connections.
- **Always** wear a heel strap when working in an area with a conductive floor. If you are uncertain about the conductivity of your floor, wear a heel strap.
- **Always** ground yourself before you clean, inspect, or make a connection to a static-sensitive device or test port. You can, for example, grasp the grounded outer shell of the test port or cable connector briefly.
- **Always** ground the center conductor of a test cable before making a connection to the analyzer test port or other static-sensitive device. This can be done as follows:
 1. Connect a short (from your calibration kit) to one end of the cable to short the center conductor to the outer conductor.
 2. While wearing a grounded wrist strap, grasp the outer shell of the cable connector.
 3. Connect the other end of the cable to the test port and remove the short from the cable.

See [Analyzer Accessories](#) for ESD part numbers.

Absolute Output Power

An absolute output-power measurement displays absolute power versus frequency.

- [What is Absolute Output Power?](#)
- [Why Measure Absolute Output Power?](#)
- [Accuracy Considerations](#)
- [How to Measure Absolute Output Power](#)

[See other Amplifier Parameters topics](#)

What is Absolute Output Power?

An absolute-output power measurement displays the power present at the analyzer's input port. This power is absolute-it is not referenced (ratioed) to the incident or source power. In the log mag format, values associated with the grid's vertical axis are in units of dBm, which is the power measured in reference to 1 mW.

- 0 dBm = 1 mW
- -10 dBm = 100 μ W
- +10 dBm = 10 mW

In the linear mag format, values associated with the grid's vertical axis are in units of watts (W).

Why Measure Absolute Output Power?

Absolute output power is measured when the amplifier's output must be quantified as absolute power rather than a ratioed relative power measurement. For example, during a gain compression measurement, it is typical to also measure absolute output power. This shows the absolute power out of the amplifier where 1-dB compression occurs.

Accuracy Considerations

The output power of the amplifier should be sufficiently attenuated if necessary. Too much output power could:

- Damage the analyzer receiver

- o Exceed the input compression level of the analyzer receiver, resulting in inaccurate measurements.

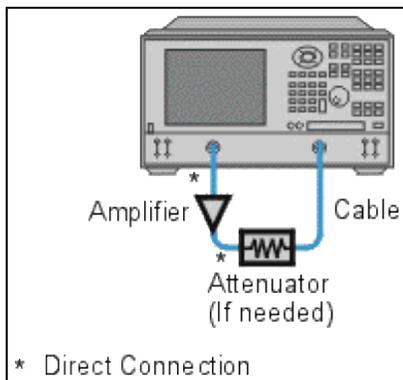
Attenuation of the amplifier's output power can be accomplished using either attenuators or couplers

The amplifier may respond very differently at various temperatures. The tests should be done when the amplifier is at the desired operating temperature.

How to Measure Absolute Power

Do the following to measure absolute output power:

1. Preset the analyzer.
2. Select an unratiod power measurement (receiver B). [Learn how.](#)
3. Set the analyzer's source power to 0 dBm.
4. Select an external attenuator (if needed) so the amplifier's output power will be sufficiently attenuated to avoid causing receiver compression or damage to the analyzer's port-2.
5. Connect the amplifier as shown in the following graphic, and provide the dc bias.



6. Select the analyzer settings for your amplifier under test.
7. Remove the amplifier and connect the measurement ports together. Store the data to memory. Be sure to include the attenuator and cables in the test setup if they will be used when measuring the amplifier.
8. Save the instrument state to memory.
9. Reconnect the amplifier.
10. Select the data math function Data/Memory.
11. Scale the displayed measurement for optimum viewing and use a marker to measure the absolute output-power at a desired frequency.
12. Print or save the data to a disk.

Last modified:

25-Nov-2013 Updated with link for MM.

Active Probing with the PNA

You can use passive and active, single and differential probes with the PNA.

RF Probes are available from Agilent, including the U1818A/B active probe with a maximum frequency of 12 GHz. [Learn more at the Agilent website.](#)

Note: The PNA does NOT have a probe power port.

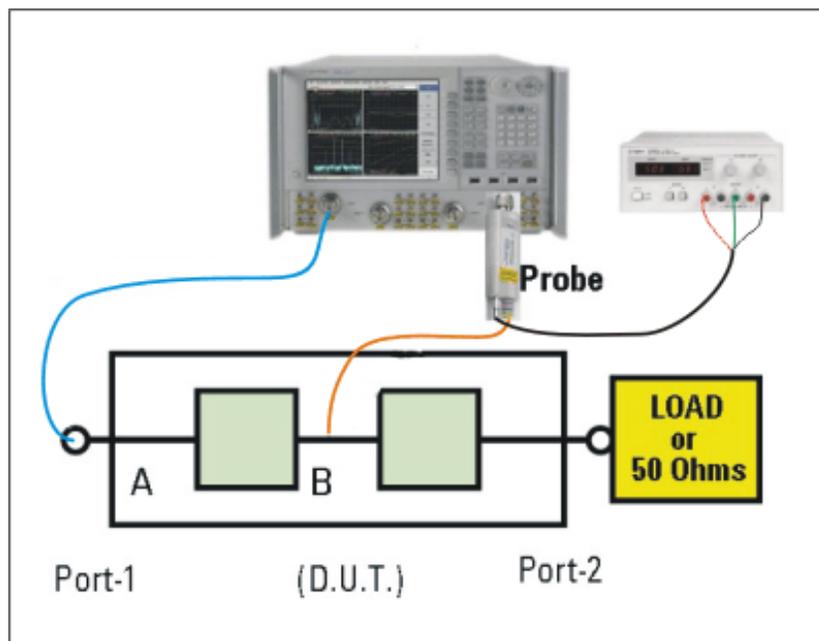
Therefore, order the U1818A/B with banana plugs (Opt 002) for powering with an external power supply, such as the Agilent E3620A Dual Output Power Supply or equivalent.

See Also

- [App Note 5990-4387EN](#): High Frequency Probing Solutions for Time and Frequency Domain Applications.
- [U1818A/B Technical Overview](#)

Procedure

Use the following general procedure to make PNA measurements with ONE U1818B active probe:



1. Connect the U1818A/B banana plugs to an external power supply. Power requirements for the U1818A/B are **+15V** (at 42mA) and **-12.6V** (at 12 mA).
2. Connect the U1818A/B to the PNA port 2 directly, or using an adapter or short cable.

3. Connect your DUT input connector to the PNA port 1 directly, or using an adapter or short cable.
4. On the PNA, press **Preset**.
5. Press **Meas**, then **S21** (transmission measurement).
6. Press **Freq**, then select the Start and Stop frequency range of the measurement. The maximum frequency of the U1818B is 12 GHz.
7. Connect the probe tips as close as possible to the DUT input connector (point A in the above image).

Note: The probe tip has two pins:

- One pin is connected to the signal trace.
- The other pin is connected to the ground trace.

8. Press **Cal**, then **Start Cal**, then **Cal Wizard**.
9. Select **Unguided** and **Next>**, then **Response** and **Next>**, then **Normalize** and **Next>** then **Finish**.
 - This calibration removes the losses (from the PNA test ports to the DUT) from subsequent measurements. [Learn more about PNA calibration.](#)
 - When correctly calibrated, the S21 measurement should show a flat response at 0 dB across the frequency range.
 - Connect the probe tips anywhere in the DUT path to view the frequency response between the DUT input and the probe tips.
 - To view the response in Time Domain, press **Analysis**, then **Transform**, then **Transform ON**. Learn about [Time Domain measurements](#).

Last modified:

23-Feb-2012 New topic

AM-PM Conversion

The AM-PM conversion of an amplifier is a measure of the amount of undesired [phase deviation \(PM\)](#) that is caused by amplitude variations (AM) inherent in the system.

- [What Is AM-PM Conversion?](#)
- [Why Measure AM-PM Conversion](#)
- [Accuracy Considerations](#)
- [How to Measure AM-PM Conversion](#)

Other Tutorials topics

What Is AM-PM Conversion?

AM-to-PM conversion measures the amount of undesired phase deviation (PM) that is caused by amplitude variations (AM) of the system. For example, unwanted phase deviation (PM) in a communications system can be caused by:

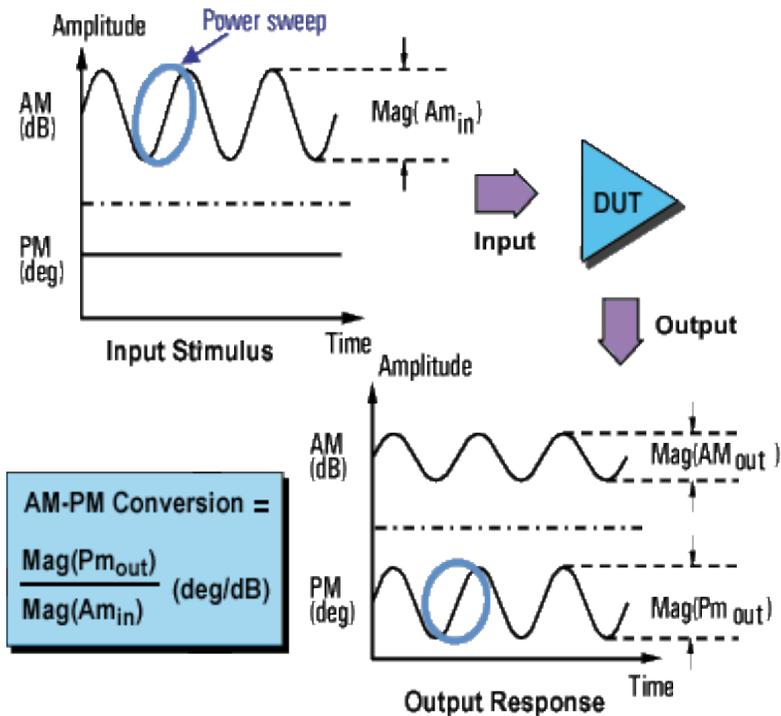
Unintentional amplitude variations (AM)

- Power supply ripple
- Thermal drift
- Multipath fading

Intentional modulation of signal amplitude

- QAM
- Burst modulation

AM-to-PM conversion is usually defined as the change in output phase for a 1-dB increment in the power-sweep applied to the amplifier's input (i.e. at the 1 dB gain compression point). It is expressed in degrees-per-dB ($^{\circ}/\text{dB}$). An ideal amplifier would have no interaction between its phase response and the power level of the input signal.



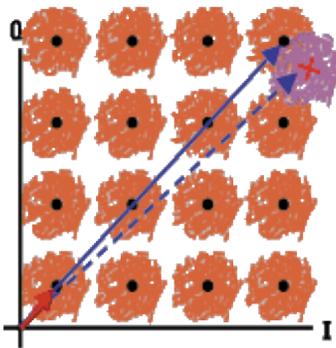
Why Measure AM-PM Conversion

AM-to-PM conversion is a critical parameter in systems where **phase** (angular) modulation is used, such as:

- FM
- QPSK
- 16QAM

It is a critical parameter because undesired phase deviation (PM) causes analog signal degradation, or increased bit-error rates (BER) in digital communication systems. While it is easy to measure the BER of a digital communication system, this measurement alone does not help you understand the underlying causes of bit errors. AM-to-PM conversion is one of the fundamental contributors to BER, and therefore it is important to quantify this parameter in communication systems.

Refer to the I/Q diagram below for the following discussion on how AM-to-PM conversion can cause bit errors.



AM to PM conversion
can cause bit errors

- The desirable state change is from the small solid vector to the large solid vector.
- With AM-to-PM conversion, the large vector may actually end up as shown with the dotted line. This is due to phase shift that results from a change in the input power level.
- For a 64QAM signal as shown (only one quadrant is drawn), we see that the noise circles that surround each state would actually overlap, which means that statistically, some bit errors would occur.

Accuracy Considerations

With this method of measuring AM-to-PM conversion, the modulation frequency is approximately the inverse of the sweep time. Even with the fastest power sweep available on most network analyzers, the modulation frequency ends up being fairly low (typically less than 10 Hz). This could cause a slight temperature change as the sweep progresses, especially if the amplifier has low thermal mass, typical of an unpackaged device. Results using this method could differ slightly if the nonlinear behavior of an amplifier is extremely sensitive to thermal changes. (The PNA series analyzers can make power sweeps <1 ms.)

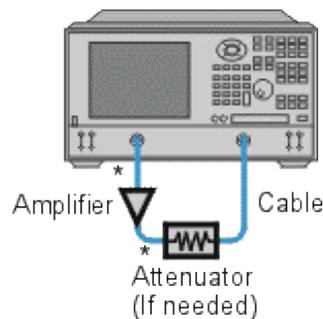
- The amplifier may respond very differently at various temperatures. The tests should be done when the amplifier is at the desired operating temperature.
- The output power of the amplifier should be sufficiently attenuated if necessary. Too much output power could:
 - damage the analyzer receiver
 - exceed the input compression level of the analyzer receiver, resulting in inaccurate measurements
- **Attenuation** of the amplifier's output power can be accomplished using:
 - Attenuators

- Couplers

- The frequency-response effects of the attenuators and couplers must be accounted for during **calibration** since they are part of the test system. Proper error-correction techniques can reduce these effects.
- The **frequency response** is the dominant error in an AM-to-PM conversion measurement setup. Performing a thru-response measurement calibration significantly reduces this error. For greater accuracy, perform a 2-port measurement calibration.

How to Measure AM-PM Conversion

1. Preset the analyzer.
2. Select an S21 measurement in the power-sweep mode.
3. Enter the start and stop power levels for the analyzer's power sweep. The start power level should be in the linear region of the amplifier's response (typically 10-dB below the 1-dB compression point). The stop power level should be in the compression region of the amplifier's response.
4. Select an external attenuator (if needed) so the amplifier's output power will be sufficiently attenuated to avoid causing receiver compression or damage to the analyzer's port 2.
5. Connect the amplifier as shown in the following graphic, and provide the dc bias.



* Direct Connection

6. Select the analyzer settings for your amplifier under test in order to perform a swept-power gain compression measurement at a chosen frequency. See [Gain Compression](#).
7. Remove the amplifier and perform a measurement calibration. Be sure to include the attenuator and cables in the calibration setup if they will be used when measuring the amplifier.
8. Save the instrument state to memory.
9. Reconnect the amplifier.
10. Use a reference marker to target the amplifier's input power at the 1-dB gain compression point. Select a second marker and adjust its stimulus value until its response is 1-dB below the reference marker.
11. Change the S₂₁ measurement from a log magnitude format to a **phase** format (no new calibration is

required).

12. Find the **phase** change between the markers. The value is the AM-to-PM conversion coefficient at the 1-dB gain compression point.
13. Print the data or save it to a disk.

Amplifier Parameters Reference

- [Gain](#)
- [Gain Flatness](#)
- [Reverse Isolation](#)
- [Gain Drift Versus Time](#)
- [Deviation from Linear Phase](#)
- [Group Delay](#)
- [Return Loss \(SWR, \$\rho\$ \)](#)
- [Complex Impedance](#)
- [Gain Compression](#)
- [AM-to-PM Conversion](#)

See Also

- [High-Gain Amplifiers](#)
- [High Power with PNA-X](#)

Gain

$$\tau = \frac{V_{\text{trans}}}{V_{\text{inc}}}$$
$$\text{Gain (dB)} = -20 \log_{10} |\tau|$$
$$\text{Gain (dB)} = P_{\text{out}} \text{ (dBm)} - P_{\text{in}} \text{ (dBm)}$$

The ratio of the amplifier's output power (delivered to a Z_0 load) to the input power (delivered from a Z_0 source). Z_0 is the [characteristic impedance](#), in this case, 50Ω .

For small signal levels, the output power of the amplifier is proportional to the input power. Small signal gain is the gain in this linear region.

As the input power level increases and the amplifier approaches saturation, the output power reaches a limit and the gain drops. Large signal gain is the gain in this nonlinear region. See [Gain Compression](#).

Gain Flatness

The variation of the gain over the frequency range of the amplifier. See [Small Signal Gain and Flatness](#).

Reverse Isolation

The measure of transmission from output to input. Similar to the gain measurement except the signal stimulus is applied to the output of the amplifier. See [Reverse Isolation](#).

Gain Drift versus Time (temperature, bias)

The maximum variation of gain as a function of time, with all other parameters held constant. Gain drift is also observed with respect to other parameter changes such as temperature, humidity or bias voltage.

Deviation from Linear Phase

The amount of variation from a linear phase shift. Ideally, the phase shift through an amplifier is a linear function of frequency. See [Deviation from Linear Phase](#).

Group Delay

$$\begin{aligned}\tau_g (\text{sec}) &= - \frac{\Delta \theta}{\Delta \omega} \\ &= - \frac{1}{360} * \frac{\Delta \theta}{\Delta f}\end{aligned}$$

The measure of the transit time through the amplifier as a function of frequency. A perfectly linear phase shift would have a constant rate of change with respect to frequency, yielding a constant group delay. See [Group Delay](#).

Return Loss (SWR, ρ)

$$\begin{aligned}\Gamma &= \frac{V_{\text{refl}}}{V_{\text{inc}}} = \rho \angle \theta \\ \text{Reflection coefficient} &= \rho \\ \text{Return loss (dB)} &= -20 \log_{10} \rho \\ \text{SWR} &= \frac{1+\rho}{1-\rho}\end{aligned}$$

The measure of the reflection mismatch at the input or output of the amplifier relative to the system Z_0 characteristic impedance.

Complex Impedance

$$Z = \frac{1+\Gamma}{1-\Gamma} * Z_0$$
$$= -R + jX$$

Complex **impedance** (1+G). The amount of reflected energy from an amplifier is directly related to its impedance. Complex impedance consists of both a resistive and a reactive component. It is derived from the characteristic impedance of the system and the reflection coefficient. See [Complex Impedance](#).

Gain Compression

See [Gain Compression](#) Application.

AM-to-PM Conversion Coefficient

$$\text{AM/PM} = \frac{\Delta \theta}{\Delta P}$$

The amount of **phase** change generated in the output signal of an amplifier as a result of an amplitude change of the input signal.

The AM-to-PM conversion coefficient is expressed in units of degrees/dB at a given power level (usually P_{1dB}, which is the 1 dB gain compression point). See [AM-PM Conversion](#).

Antenna Measurements

This topic describes how to setup the PNA to make S21 measurements on an array of antennas. Measurements can be made on up to 100 antenna arrays (Ports) and up to 15 discrete frequencies

Measurement Sequence

1. The PNA is set to a start frequency.
2. As the antenna moves, the PNA responds to each external trigger signal by measuring an antenna port.
3. When all ports are measured, the PNA increments to the next frequency
4. Again the PNA measures all ports, and so forth until all ports are measured at all frequencies in the forward direction.
5. As the antenna begins moving in the opposite direction, the same sequence occurs, except the PNA decrements in frequency until all ports are measured at all frequencies and the PNA is set back to the original start frequency.

Once setup, only external trigger signals are sent to the PNA. After each trigger, measurement data is stored in internal PNA memory.

How to set up the PNA

1. On the **System** menu click **Preset**
2. On the **Sweep** menu point to **Trigger** then click **Trigger**
3. In Trigger Source click **External**
4. In Trigger Scope click **Channel**
5. Click **OK**

Forward Sweep

1. On the **Trace** menu click **New Trace**
2. Click **S21** then Channel Number **1**
3. On the **Sweep** menu point to **Trigger** then click **Trigger**
4. In Channel Trigger State check **Point Sweep**
5. Click **OK**
6. On the Sweep menu click **Sweep Type**:then **Segment Sweep**

7. Click **OK**
8. On the **View** menu point to **Tables** then click **Segment Table**
9. Do this 15 times - Sweep menu point to **Segment Table** then **Insert Segment**
10. For each Segment in the Segment table:
 1. Click **State**:and select **ON**
 2. Double click both **START** and **STOP** Frequency: (each new segment ascends in frequency)
 3. Double click **Points**: type Number of Ports (elements)

Reverse sweep

Repeat the following steps for each frequency: (up to 15)

- Increment the channel number (**X**) Starting with Channel 2
 - Decrement the frequency (**F**)
1. On the **Trace** menu click **New Trace...**
 2. Click **S21** then Channel Number **X**
 3. When a window contains four traces, check **Create in New Window**.
 4. Click **OK**
 5. On the **Sweep** menu point to **Trigger** then click **Trigger**
 6. In Channel Trigger State check **Point Sweep**
 7. Click **OK**
 8. On the Sweep menu click **Sweep Type**:then **Segment Sweep**
 9. Click **OK**
 10. On the **View** menu point to **Tables** then click **Segment Table**
 11. In the Segment table
 1. Click **State**:and select **ON**
 2. Double click both **START** and **STOP** Frequency **F**
 3. Double click **Points**: type Number of Ports (elements)
-

Last Modified:

8-Apr-2008 Removed reference to antenna macro

Balanced Measurements

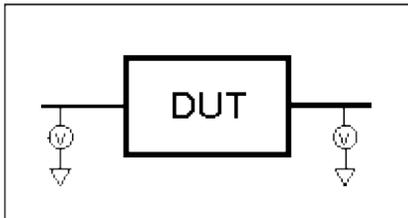
- [What are Balanced Devices?](#)
- [Differential and Common Modes Model](#)
- [Measuring Mixed Mode \(Balanced\) S-Parameters](#)
- [Measuring Imbalance Parameters](#)
- [Measuring CMRR](#)
- [Port Mapping](#)
- [Calibrating Balanced Measurements](#)
- [How the PNA makes Balanced Measurements](#)

Other [Measurement Setup](#) Topics

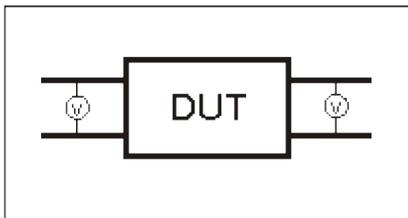
Check out the **Integrated True Mode Stimulus Application** ([iTMSA](#)).

What are Balanced Devices?

Standard **Single-ended devices** generally have one input port and one output port. Signals on the input and output ports are referenced to ground.



Balanced devices have two pins on either the input, the output, or both. The signal of interest is the difference and average of the two input or output lines, not referenced to ground.



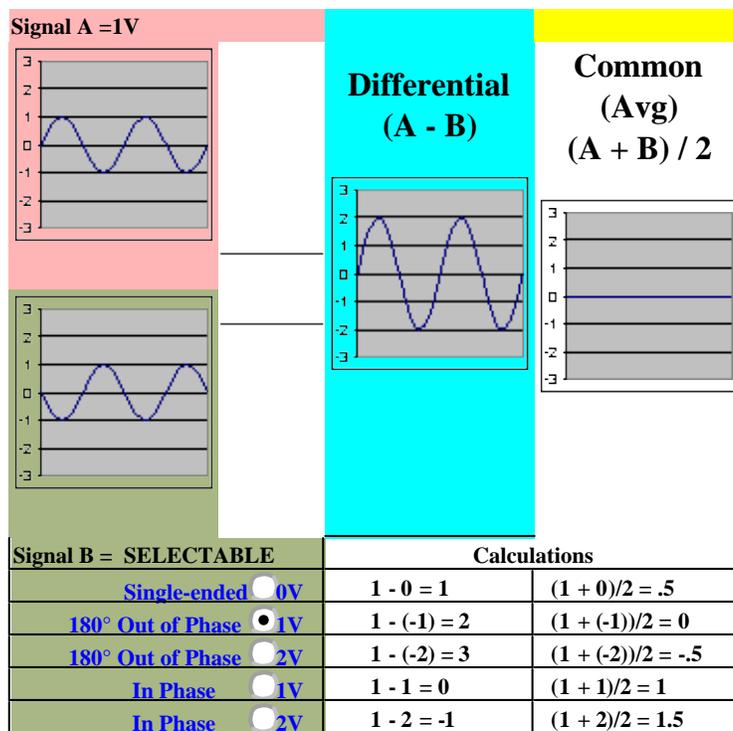
Differential and Common Modes Model

On balanced devices, the signal of interest is the **difference** and **average** of the two input or output lines. In balanced device terminology, these signals are known as the Differential and Common modes.

The following model shows how two signals (A and B) combine to create Differential and Common mode signals:

- **Signal A** is fixed at 1V peak
- **Signal B** is selectable
- **Differential** is calculated as **A minus B**
- **Common** is calculated as the **AVERAGE** of **A and B**

Note: Click **Signal B** selections to see various Differential and Common signals.



Notes:

- Even when Signal B is 0V, like a Single-ended signal, there is still a unique Differential and Common mode representation of the two individual signals.
- The above model does not show a DUT. The difference and average of two signals can be calculated for both the balanced INPUT and balanced OUTPUT of a device.

Measuring Mixed Mode (Balanced) S-Parameters

Mixed mode S-parameters combine traditional S-parameter notation with balanced measurement terminology.

Some balanced devices are designed to amplify the differential component and reject the common component. This allows noise that is common to both inputs to be virtually eliminated from the output. For example, a balanced device may amplify the differential signal by a factor of 5, and attenuate the common signal by a factor of 5. Using traditional S-parameter notation, an S21 is a ratio measurement of the device **Output** / device **Input**. Mixing this with balanced terminology, we could view the amplifier's Differential Output signal / Differential Input signal. To see this parameter on the PNA, we would select an Sdd21 measurement using the following balanced notation:

Sabxy -

Where

a - device output mode

b - device input mode

(choose from the following for both a and b:)

- **d** - differential
- **c** - common
- **s** - single ended

x - device output "logical" port number

y - device input "logical" port number

See Also

[Logical port mapping](#)

[Port mapping with External Test Sets](#)

[iTMSA](#)

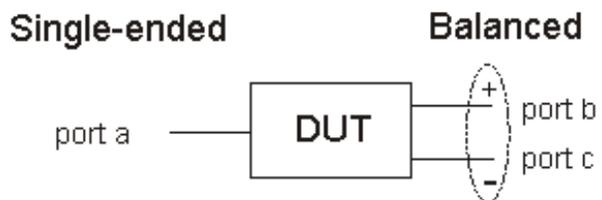
Measuring Imbalance Parameters

Imbalance is a measure of how well two physical ports that make up a balanced port are matched. With a perfectly balanced port, the same amount of energy flows to both ports and the magnitude of the ratio of these ports is 1.

The notation is similar to traditional S-parameters. In the following diagrams, the letters a, b, c, and d are used because any PNA port can be assigned to any logical port using the [port mapping process](#).

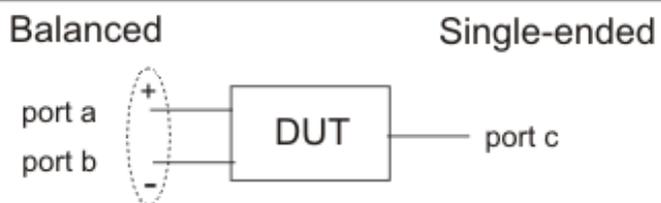
For example, in the following single-ended - balanced formula, **Sba** indicates the device output port is logical port b and the input port is logical port a.

Imbalance parameter when measuring a **single-ended - balanced** device.



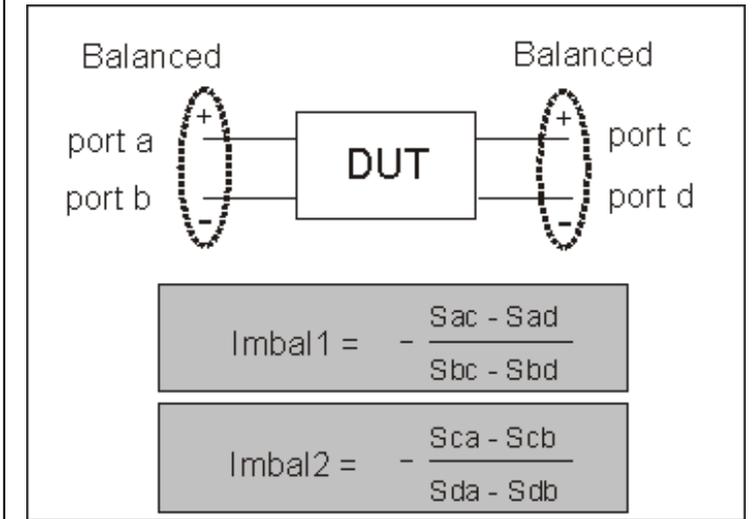
$$\text{Imbal} = - \frac{S_{ba}}{S_{ca}}$$

Imbalance parameter when measuring a **balanced - single-ended** device.

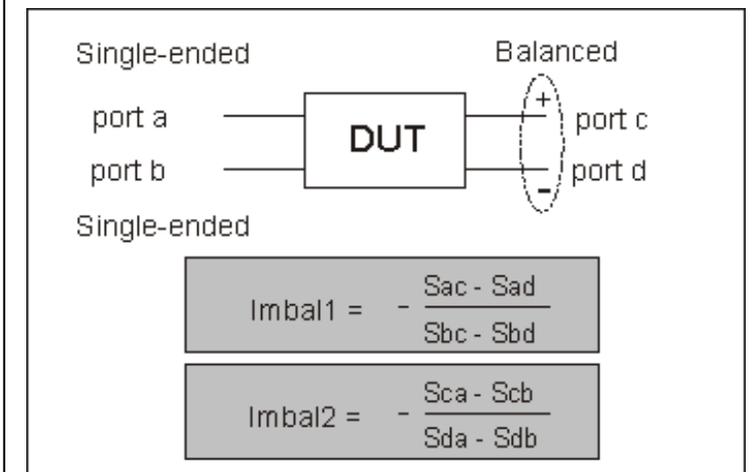


$$\text{Imbal} = - \frac{S_{ca}}{S_{cb}}$$

Imbalance1 and Imbalance2 parameters when measuring a **balanced - balanced** device.



Imbalance1 and Imbalance2 parameters when measuring a **single-ended - single-ended - balanced** device.



Measuring CMRR (Common Mode Rejection Ratio)

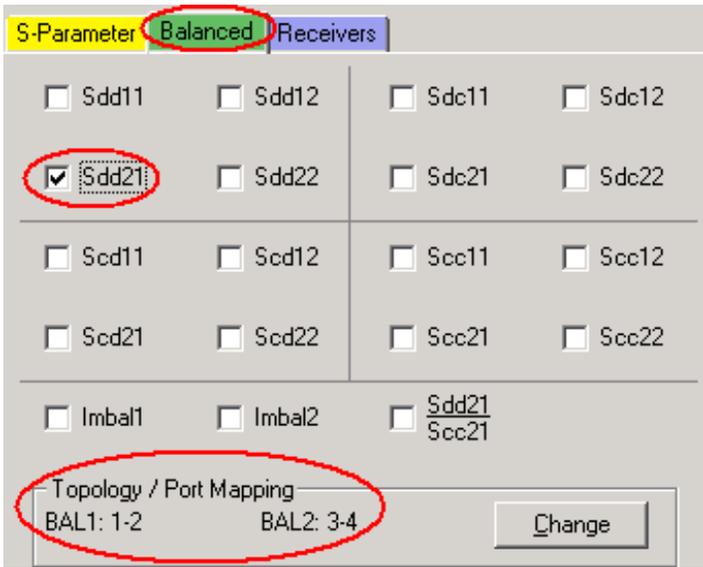
CMRR is a ratio of the transmission characteristic in differential mode over the transmission characteristic in the common mode of the balanced port as the measurement parameter. A high value indicates more rejection of common mode, which is desirable in a device that transmits information in the differential portion of the signal. The table below shows the CMRR parameter you can select when measuring each balanced device.

Single-ended - balanced device	Sds21 ----- Scs21	and	Ssd12 ----- Ssc12
Balanced - single-ended device	Ssd21 ----- Ssc21	and	Sds12 ----- Scs12
Balanced - balanced device	Sdd21 ----- Scc21		
Single-ended - single-ended - balanced device	Sds31 ----- Scs31	and	Sds32 ----- Scs32

Device Topology and Port Mapping

As we have seen on balanced inputs and outputs, the signal of interest is the difference or average of two BALANCED input or BALANCED output lines. It is also possible to have single-ended ports AND balanced ports on the same device. The two balanced input or output lines are referred to as a single "logical" port.

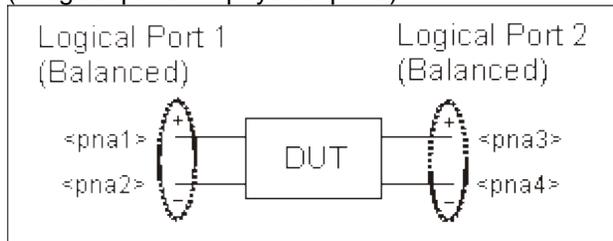
When configuring a balanced measurement on the PNA, select a device 'topology'. Then map each PNA test port to the DUT ports. The PNA assigns "logical ports". [See how to set device topology in the PNA.](#)



The following device topologies can be measured by a 4-port PNA.

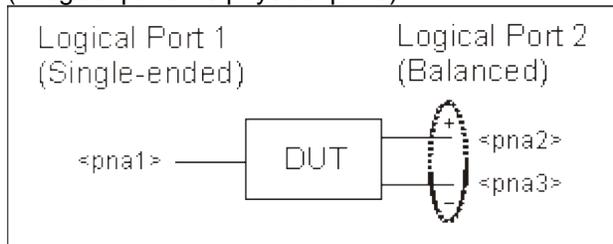
- **Balanced / Balanced**

(2 logical ports - 4 physical ports)



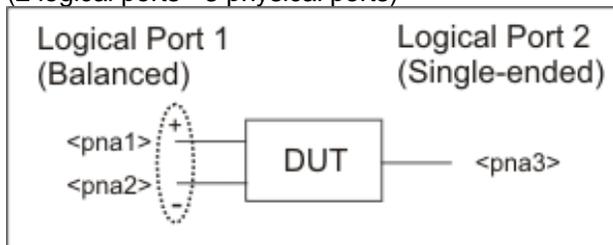
- **Single-ended / Balanced**

(2 logical ports - 3 physical ports)



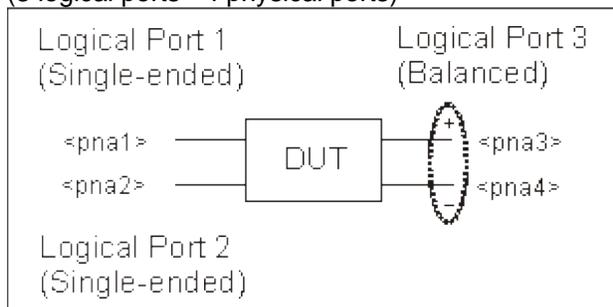
- **Balanced / Single-ended**

(2 logical ports - 3 physical ports)



- **Single-ended - Single-ended / Balanced**

(3 logical ports - 4 physical ports)



These topologies can be used in the reverse (\Leftarrow) direction to measure:

- **Balanced / Single-ended** topology
- **Balanced / Single-ended - Single-ended** topology

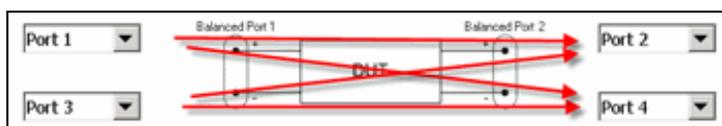
For example, to measure a **Balanced / Single-ended** topology, measure the S12 (reverse direction) of a **Single-ended / Balanced** topology.

Calibrating Balanced Measurements

Balanced measurements are calibrated in the same manner as single-ended (standard) measurements. However, for highest accuracy, you must choose Thru paths so that each **transmission path** of the balanced measurement is represented. For a Balanced/Balanced topology, this means that FOUR Thru connections should be made.

For example (see following image):

- Balanced Port 1 is ports 1 and 3
- Balanced Port 2 is ports 2 and 4
- Thru paths to be calibrated should be: 12, 14, 32, 34.
- Paths 13, and 24 are less important.



To select Thru paths:

1. From SmartCal, on the Select DUT Connectors and Cal Kits page, check **Modify Cal**.
2. Click **Next** to see the following Cal Wizard page:

	1st Port	2nd Port	Thru Cal Method	
Thru #1	1	2	Unknown Thru	Cal Type/Std...
Thru #2	1	4	Unknown Thru	Cal Type/Std...
Thru #3	3	4	Unknown Thru	Cal Type/Std...
Thru #4	2	3	Unknown Thru	Cal Type/Std...

How the PNA makes Balanced Measurements

When using standard Balanced measurements, the PNA does not provide true balanced measurements by stimulating both balanced inputs together and measuring both outputs relative to one another. Instead, the PNA makes only Single-ended measurements. On a Balanced/ Balanced device, it stimulates each input and measures each output individually. From the output data, the PNA calculates the Differential and Common outputs from the DUT using the same math formulas as the above model. However, all measurements and calculations on the PNA are performed in frequency domain using complex (magnitude and phase) data. The Balanced S-parameter display data is then calculated from the Differential and Common inputs and outputs.

In [iTMSA](#), the PNA **DOES** stimulate both balanced inputs with true balanced sources.

Last Modified:

25-Apr-2012 Added BalSe (9.70)

2-Aug-2011 Added Calibrating

15-May-2008 Edited for iTMSA

Complex Impedance

When making an S_{11} or S_{22} measurement of your device under test, you can view complex-impedance data such as series resistance and reactance as well as [phase](#) and magnitude information. Complex impedance data can be viewed using either the Smith Chart format or the Polar format.

- [What Is Complex Impedance?](#)
- [Accuracy Considerations](#)
- [How to Measure Complex Impedance](#)

What Is Complex Impedance?

Complex-impedance data is information that can be determined from an S_{11} or S_{22} measurement of your device under test, such as:

- Resistance
- Reactance
- Phase
- Magnitude

The amount of power reflected from a device is directly related to the impedances of both the device and the measuring system. For example, the value of the complex reflection coefficient (Γ) is equal to 0 only when the device impedance and the system impedance are exactly the same (i.e. maximum power is transferred from the source to the [load](#)). Every value for Γ corresponds uniquely to a complex device impedance (as a function of frequency), according to the equation:

$$Z_L = [(1 + \Gamma) / (1 - \Gamma)] \times Z_0$$

where Z_L is your test device impedance and Z_0 is the measuring system's [characteristic impedance](#).

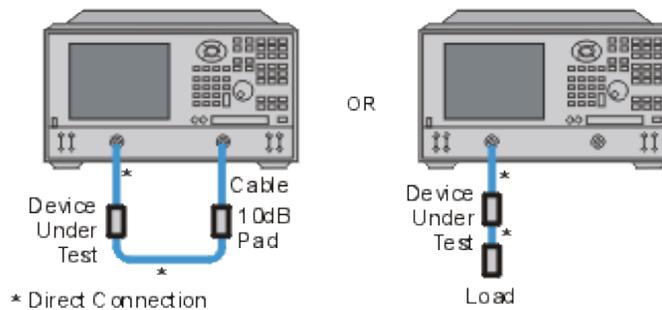
Complex Impedance is best viewed using either [Polar](#) or [Smith Chart](#) format.

Accuracy Considerations

- The Smith chart is most easily understood when used with a full scale value of 1.0.
- For greater accuracy when using markers in the Smith chart or polar formats, activate the discrete marker mode.

- The uncertainty of reflection measurements is affected by:
 - Directivity
 - Reflection tracking
 - Source match
 - Load match (with 2-port devices)

With a 2-port **calibration**, the effects of these factors are reduced. A 1-port calibration provides the same accuracy if the output of the device is well terminated. Refer to the graphic below for the following discussion.



- If you connect the device between both analyzer ports, it is recommended that you use a 10 dB pad on the output of the device to improve measurement accuracy. This is not necessary if you use a 2-port calibration since it corrects for load match.
- If you connect a two-port device to only one analyzer port, it is recommended that you use a high-quality **load** (such as a calibration standard) on the output of the device.

How to Measure Complex Impedance

1. Connect the device as shown in the previous graphic.
2. Preset the analyzer.
3. Set up, calibrate, and perform an S11 or S22 measurement.
4. View impedance data:
 - a. Select the Smith Chart format.
 - b. Scale the displayed measurement for optimum viewing.
 - c. Position the marker to read the resistive and reactive components of the complex impedance at any point along the trace.
 - d. Print the data or save it to a disk.

5. View the magnitude and **phase** of the reflection coefficient:

- a. Select the Smith chart format or the Polar format.
- b. Select either Lin Marker or Log Marker formats.
- c. Scale the displayed measurement for optimum viewing.
- d. Position the marker to read the frequency, magnitude, and phase of the reflection coefficient (Γ) at any point along the trace.
- e. Print the data or save it to a disk.

Comparing the PNA "Delay" Functions

The PNA has three Delay functions which are similar but are used in different ways.

1. **Group Delay format** is used to display the Group Delay of a network. Group Delay is defined as:

$$-d(\phi)/d(\omega) \text{ -- where } \phi \text{ is radian angle, and } \omega \text{ is radian frequency.}$$

Since it is defined by a derivative, the value must be determined from an analytic function. However, the PNA makes discrete measurements, so we approximate the group delay by taking the finite difference:

$$-(1/360)*\Delta(\phi)/\Delta(f) \text{ -- where } \phi \text{ is degree angle and } f \text{ is frequency in Hz. The } 1/360 \text{ does the proper conversion of degrees to radians and Hz frequency to radian frequency.}$$

From this we can see that, if the phase response of a network varies with frequency, then the Group Delay must vary as well. In fact, many filters are specified by the variation of their Group Delay.

If we measure the phase response of a lossless cable, it should be a straight line. But, of course, nothing is perfect. The phase response will have a small amount of noise. This is due to trace noise of the PNA, and the loss with real cables or transmission lines, which causes a small amount of non-linear phase change with frequency. So, if we look at the Group Delay of a cable, we will see a small amount of variation. Also, if the frequency spacing is small enough when you make the measurement, the $\Delta(f)$ in the denominator becomes very small, so the delay can have wide swings with just a little noise.

To overcome this issue, we sometimes add smoothing to a phase trace, which widens the effective $\Delta(f)$, called the aperture, and provides a less noisy Group Delay response. The Group Delay of a device is only valid for a given frequency aperture. [Learn more about Group Delay.](#)

2. **Electrical Delay** function. On many filters, the passband response is specified for a maximum value of "Deviation from Linear Phase". When looking at the passband of a multi-pole filter, one sees the phase changing very rapidly. This makes it difficult to determine the linearity of the phase response. The Electrical Delay function subtracts out a "LINEAR PHASE" equivalent to the delay time value computed as above. When you use this function, you dial in the Linear Delay such that a CONSTANT PHASE SLOPE is removed from the phase trace, until the phase trace is mostly flat. The remaining variation is the deviation from linear phase.

To make this task a little less tedious, the PNA has a marker function called [Marker ==>> Delay](#). This function computes the Group Delay value at the marker position, using a 20% smoothing aperture, then changes the Electrical Delay value to this value. Obviously, if the phase trace is not perfectly linear, moving the marker and recomputing the delay will result in different values. The phase slope added by

the electrical delay function applies only to the current measurement. That is, each measurement (S11, S22, S12, S21) can have its own value of electrical delay. [Learn more about Deviation from Linear Phase.](#)

3. [Port Extension](#) is a function that is similar to calibration. It applies to all the traces in a given channel. It compensates for the phase response change that occurs when the calibration reference plane is not the same as the measurement plane of the device.

Let's look at an example of a DUT that is mounted on a PCB fixture with SMA connectors. We can easily calibrate at the SMA connectors. But if we add the fixture to measure the board-mounted device, the apparent phase of the DUT is changed by the phase of the PCB fixture. We use port extensions to add a LINEAR PHASE (constant delay) to the calibration routines to shift the phase reference plane to that of the DUT. This is ONLY valid if the fixture consists of a transmission line with linear phase response, and this limitation is usually met in practice. The main reason that it is NOT met is that there is mismatch at the SMA-to-PCB interface. This mismatch was not removed with the error correction because it occurs AFTER the SMA connector. Ripple can be seen on the display as signals bounce back and forth between the mismatch and the DUT. If the DUT is well matched, the ripple effect is very small. However, when we use Automatic Port Extension (APE), and we leave the fixture open (the DUT removed), the reflection is large and we see larger ripples. That is why APE uses a curve fitting process to remove the effects of the ripple. For best effect, the wider the IF Bandwidth, the better we can "smooth-out" the ripples with curve fitting. Still, we are fitting a LINEAR PHASE SLOPE to the phase response, and thus we use only a single Port Extension Delay value to represent the phase slope.

The method used by older VNAs to get this same functionality was to add a mechanical line stretcher to the reference channel, which removed a fixed delay amount from the port. Port extensions give 1x the delay for transmission at each port, and 2x the delay for reflection, so it differs somewhat from Electrical Delay above, in that the math function depends upon the measurement being made. The signal passes twice through the fixture for reflection (out and back), but only once for each port on transmission. For S21, the phase slope added is the sum of the port 1 and port 2 Port Extension Delay values.

The "User Range" APE function is used in cases where a fixture has limited bandwidth, perhaps due to tuning elements or bias elements. In this case, the model of constant delay for the fixture over the whole bandwidth is not valid, so a narrower "User Range" of frequencies can be selected to compute the delay. Since the aperture is smaller, there is more uncertainty in the delay computation for port extension. Also, for those who had been using the [Marker ==>> Delay](#) function to estimate the delay, we added the "Active Marker" selection to APE, which works exactly the same as Marker->Delay. [Learn more about Automatic Port Extensions.](#)

Deviation from Linear Phase

Deviation from linear phase is a measure of phase distortion. The electrical delay feature of the analyzer is used to remove the linear portion of the phase shift from the measurement. This results in a high-resolution display of the non-linear portion of the phase shift (deviation from linear phase).

- [What Is Linear Phase Shift?](#)
- [What Is Deviation from Linear Phase?](#)
- [Why Measure Deviation from Linear Phase?](#)
- [Using Electrical Delay](#)
- [Accuracy Considerations](#)

See also [Comparing the PNA Delay Functions](#)

See other Tutorials

What Is Linear Phase Shift?

Phase shift occurs because the wavelengths that occupy the electrical length of the device get shorter as the frequency of the incident signal increases. *Linear* phase-shift occurs when the phase response of a device is linearly proportional to frequency. Displayed on the analyzer, the phase-versus-frequency measurement trace of this ideal linear phase shift is a straight line. The slope is proportional to the electrical length of the device. Linear phase shift is necessary (along with a flat magnitude response) for distortionless transmission of signals.

What Is Deviation from Linear Phase?

In actual practice, many electrical or electronic devices will delay some frequencies more than others, creating non-linear phase-shift (distortion in signals consisting of multiple-frequency components). Measuring deviation from linear phase is a way to quantify this non-linear phase shift.

Since it is only the deviation from linear phase which causes phase distortion, it is desirable to remove the linear portion of the phase response from the measurement. This can be accomplished by using the electrical delay feature of the analyzer to mathematically cancel the electrical length of the device under test. What remains is the deviation from linear phase, or phase distortion.

Why Measure Deviation from Linear Phase?

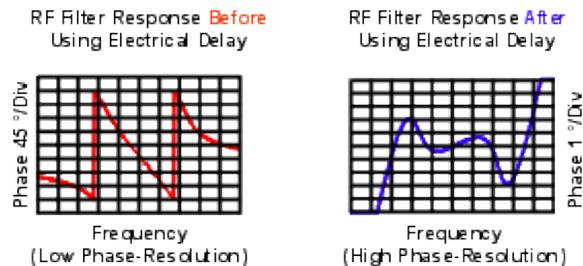
The deviation from linear phase measurement accomplishes the following:

- Presents data in units of phase rather than units of seconds (group delay). For devices that pass modulated signals, units of phase may be most practical.
- Provides a less noisy measurement than a [group delay](#) measurement.

Using Electrical Delay

The electrical delay feature is the electronic version of the mechanical "line stretcher" of earlier analyzers. This feature does the following:

- Simulates a variable-length lossless transmission line, which is effectively added to or removed from the reference signal path.
- Compensates for the electrical length of the device under test.
- Flattens the measurement trace on the analyzer's display. This allows the trace to be viewed at high resolution in order to see the details of the phase nonlinearity.
- Provides a convenient method to view the deviation from linear phase of the device under test. See the following graphic.



[Learn how to set Electrical Delay.](#)

Accuracy Considerations

The frequency response of the test setup is the dominant error in a deviation from linear phase measurement. To reduce this error, perform a 2-port measurement calibration.

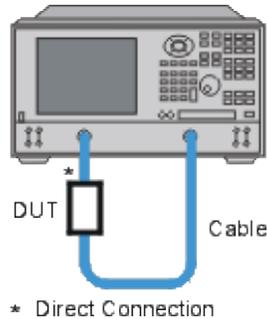
How to Measure Deviation from Linear Phase:

1. Preset the analyzer.
2. If your device under test is an amplifier, it may be necessary to adjust the analyzer's source power:
 - o Set the analyzer's source power to be in the linear region of the amplifier's output response (typically

10-dB below the 1-dB compression point).

- o Select an external attenuator (if needed) so the amplifier's output power will be sufficiently attenuated to avoid causing receiver compression or damage to the analyzer's port 2.

3. Connect the device under test as shown in the following graphic.



3. Select an S21 measurement.

4. Select the settings for your device under test, including the following:

- o [Format](#): phase
- o [Scale](#): autoscale

5. Remove the device and perform a calibration.

6. Reconnect the device.

7. Scale the displayed measurement for optimum viewing.

8. [Create a marker](#) in the middle of the trace.

9. Press the **>Delay** key to invoke the [Marker to Electrical Delay](#) function. This flattens the phase trace.

10. If desired, on the **Scale** menu, click [Electrical Delay](#) to fine-tune the flatness of the phase trace.

11. Use the markers to measure the maximum peak-to-peak deviation from linear phase.

12. Print the data or save it to a disk.

Small Signal Gain and Flatness

Small signal gain is the gain in the amplifier's linear region of operation. This is typically measured at a constant input power over a swept frequency. Gain **flatness** is the measure of the variation of gain over a specified frequency range.

- [What Is Gain?](#)
- [What Is Flatness?](#)
- [Why Measure Gain and Flatness?](#)
- [Accuracy Considerations](#)
- [How to Measure Gain and Flatness](#)

[See other Amplifier Parameter topics](#)

What Is Gain?

RF amplifier gain is defined as the difference in power between the amplifier output signal and the input signal. It is assumed that both input and output impedances of the amplifier are the same as the **characteristic impedance** of the system.

- Gain is called S_{21} using S-parameter terminology
- Gain is expressed in dB-a logarithmic ratio of the output power relative to the input power.
- Gain can be calculated by subtracting the input from the output levels when both are expressed in dBm, which is power relative to 1 milliwatt.
- Amplifier gain is most commonly specified as a minimum value over a specified frequency range. Some amplifiers specify both minimum and maximum gain, to ensure that subsequent stages in a system are not under or over driven.

What Is Flatness?

Flatness specifies how much the amplifier's gain can vary over the specified frequency range. Variations in the flatness of the amplifier's gain can cause distortion of signals passing through the amplifier.

Why Measure Small-Signal Gain and Flatness?

Deviations in gain over the **bandwidth** of interest will induce distortion in the transmitted signal because frequency components are not amplified equally. Small-signal gain allows you to quantify the amplifier's gain at a particular frequency in a 50-ohm system. Flatness allows you to view the deviations in the amplifier's gain over a specified frequency range in a 50-ohm system.

Accuracy Considerations

- The amplifier may respond very differently at various temperatures. The tests should be done when the amplifier is at the desired operating temperature.
- The output power of the amplifier should be sufficiently attenuated if necessary. Too much output power could:
 - damage the analyzer receiver
 - exceed the input compression level of the analyzer receiver, resulting in inaccurate measurements.

Attenuation of the amplifier's output power can be accomplished using:

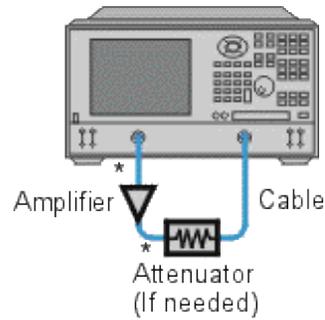
- attenuators
- couplers

The frequency-response effects and mismatches of the attenuators and couplers must be accounted for during **calibration** since they are part of the test system. Proper error-correction techniques can reduce these effects.

- The **frequency response** is the dominant error in a small-signal gain and flatness measurement setup. Performing a thru-response measurement calibration significantly reduces this error. For greater accuracy, perform a 2-port measurement calibration.
- Reducing IF bandwidth or using **averaging** improves measurement **dynamic range** and accuracy, at the expense of measurement speed.

How to Measure Gain and Flatness

1. Preset the analyzer.
2. Select an S21 measurement parameter.
3. Set the analyzer's source power to be in the linear region of the amplifier's output response (typically 10-dB below the 1-dB compression point).
4. Select an external attenuator (if needed) so the amplifier's output power will be sufficiently attenuated to avoid causing receiver compression or damage to the analyzer's port-2.



* Direct Connection

5. Connect the amplifier as shown in the following graphic, and provide the dc bias.
6. Select the analyzer settings for your amplifier under test.
7. Remove the amplifier and perform a measurement **calibration**. Be sure to include the attenuator and cables in the calibration setup if they will be used when measuring the amplifier.
8. Save the instrument-state to memory.
9. Reconnect the amplifier.
10. Scale the displayed measurement for optimum viewing and use a marker to measure the small signal gain at a desired frequency.
11. Measure the gain **flatness** over a frequency range by using markers to view the peak-to-peak ripple.
12. Print or save the data to a disk.
13. This type of measurement can be automated.

Gain Compression

Gain compression measures the level of input power applied to an amplifier that will cause a distorted output.

The [Gain Compression Application](#) (Opt 086) makes fast and accurate compression measurements.

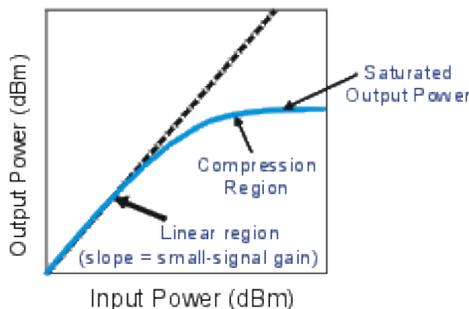
- [What Is Gain Compression?](#)
- [Why Measure Gain Compression?](#)
- [Accuracy Considerations](#)
- [How to Measure Gain Compression](#)

[See other Amplifier Parameter topics](#)

What Is Gain Compression?

Gain compression occurs when the input power of an amplifier is increased to a level that reduces the gain of the amplifier and causes a nonlinear increase in output power.

The analyzer has the ability to do power sweeps as well as frequency sweeps. Power sweeps help characterize the nonlinear performance of an amplifier. Refer to the graphic below (a plot of an amplifier's output power versus input power at a single frequency) for the following discussion.



- The amplifier has a linear region of operation where gain is constant and independent of power level. The gain in this region is commonly referred to as "small-signal gain."
- As the input power increases, the amplifier gain appears to decrease, and the amplifier goes into compression.
- The most common measurement of amplifier compression is the 1-dB compression point. This is defined as the input power (or sometimes the output power) which results in a 1-dB decrease in amplifier gain (relative to the amplifier's small-signal gain).

Why Measure Gain Compression?

When driven with a sinusoid, the output of an amplifier is no longer sinusoidal in the compression region. Some of the amplifier output appears in harmonics, rather than occurring only at the fundamental frequency of the input signal.

As input power is increased even more, the amplifier becomes saturated, and output power remains constant. At this point, further increases in amplifier input power result in no change in output power.

In some cases (such as with TWT amplifiers), output power actually decreases with further increases in input power after saturation, which means the amplifier has negative gain.

Since gain is desired in amplifier operation, it is important to know the limit of input signal that will result in gain compression.

Accuracy Considerations

The network analyzer must provide sufficient power to drive the amplifier into saturation. If you need a higher input-power level than the source of the analyzer can provide, use a preamplifier to boost the power level prior to the amplifier under test. (See [High Power PNA-X](#).) If using a preamplifier, you can increase measurement accuracy in the following ways:

- Use a coupler on the output of the preamplifier so that a portion of the boosted input signal can be used for the analyzer's reference channel. This configuration removes the preamplifier's frequency response and drift errors from the measurement (by ratioing).
- Perform a thru-response calibration including the preamplifier, couplers, and attenuators in the test setup.

The output power of the amplifier should be sufficiently attenuated if necessary. Too much output power could:

- Damage the analyzer receiver
- Exceed the input compression level of the analyzer receiver

Attenuation of the amplifier's output power can be accomplished using:

- Attenuators
- Couplers

The frequency-response effects of the attenuators and couplers must be considered during calibration since they are part of the test system. Proper error-correction techniques can reduce these effects.

- The frequency response is the dominant error in a gain compression measurement setup. Performing a thru-response measurement calibration significantly reduces this error.
- The amplifier may respond very differently at various temperatures. The tests should be done when the amplifier is at the desired operating temperature.
- Reducing IF bandwidth or using measurement averages improves accuracy, at the expense of measurement speed.

How to Measure Gain Compression

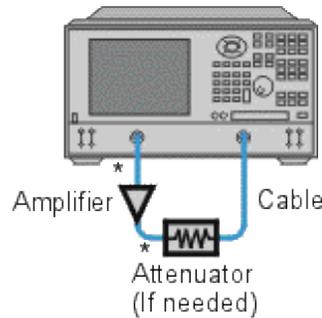
This procedure shows you how to make the following three measurements used to determine amplifier gain compression:

1. A [Swept-Frequency Gain Compression](#) measurement locates the lowest frequency at which the 1-dB gain compression first occurs.
2. A [Swept-Power Gain Compression](#) measurement shows the input power at which a 1-dB drop in gain occurs as a power ramp is applied to the amplifier at a particular frequency point (found in measurement 1).
3. An [Absolute Power](#) measurement shows the absolute power out (in dBm) at compression.

Swept-Frequency Gain Compression Measurement

A measurement of swept frequency gain compression locates the frequency point where 1-dB compression first occurs.

1. Preset the analyzer.
2. Select an S₂₁ measurement parameter.
3. Set the analyzer's source power to be in the linear region of the amplifier's output response (typically 10-dB below the 1-dB compression point).
4. Select an external attenuator (if needed) so the amplifier's output power will be sufficiently attenuated to avoid causing receiver compression or damage to the analyzer's port-2.
5. Connect the amplifier as shown in the following graphic, and provide the dc bias.
6. Select the analyzer settings for your amplifier under test. To reduce the effects of noise, you may want to specify a narrower IF bandwidth.



* Direct Connection

7. Remove the amplifier and perform a thru-response calibration. Be sure to include the attenuator and cables in the calibration setup if they will be used when measuring the amplifier.
8. Save the instrument-state to memory.
9. Reconnect the amplifier.
10. Position a marker at approximately mid-span.
11. Adjust the analyzer's scale to 1 dB per division.
12. Store the trace in memory and display Data/Mem.
13. Gradually increase the source power until a 1-dB decrease in gain is observed at the first frequency over some portion of the trace.
14. Use markers to locate the frequency where the 1-dB decrease in gain first occurs. Note this frequency for use in the following measurement.
15. Print the data or save it to a disk.

Swept-Power Gain Compression Measurement

A swept-power gain compression measurement shows the input power resulting in a 1-dB drop in gain as a power ramp at a particular frequency (found in step 13 of the previous measurement) is applied to the amplifier.

1. If not already done, perform the previous measurement of swept-frequency gain compression.
2. Setup an S_{21} measurement in the power-sweep mode. Include the following settings:
 - o Set the CW frequency to the frequency noted in step 14 of the previous measurement of swept-frequency gain compression.
 - o Enter the start and stop power levels for the sweep. The start power should be in the linear region of the amplifier's response (typically 10 dB below the 1-dB compression point). The stop power should be in the compression region of the amplifier's response.

3. Adjust the scale to 1-dB per division.
4. Use markers (including reference marker) to find the input power where the 1-dB decrease in gain occurs.
5. Print the data or save it to a disk.

Absolute Output Power Measurement

An absolute-power measurement shows the absolute power-out (in dBm) of the amplifier at compression.

1. Select an unratiod (absolute) power measurement. Choose the B input if using the test setup in the previous graphic.
2. Retain the CW frequency used in the previous measurement of swept-power gain compression.
3. Set a marker to the input power level where the 1-dB decrease in gain occurs (found in step 4 of the previous measurement).
4. Scale the displayed measurement for optimum viewing.
5. Read the marker value to find the absolute output power of the amplifier (in dBm) where the 1-dB decrease in gain occurs.
6. Print the data or save it to a disk.

Note: The measurement calibration does not apply to absolute power. Therefore, if there is any attenuation external to the analyzer, you will have to correct for it manually.

Group Delay

Group delay is a measure of phase distortion. Group delay is the actual transit time of a signal through a device under test as a function of frequency. When specifying group delay, it is important to specify the aperture used for the measurement.

- [What is Group Delay?](#)
- [Group Delay versus Deviation from Linear Phase](#)
- [What Is Aperture?](#)
- [Accuracy Considerations](#)
- [How to Measure Group Delay](#)

See also [Comparing the PNA Delay Functions](#).

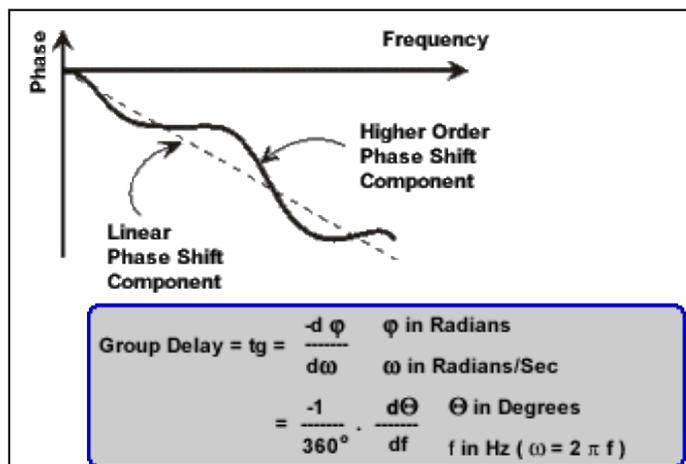
[See other Amplifier Parameter topics](#)

What Is Group Delay?

Group delay is:

- A measure of device phase distortion.
- The transit time of a signal through a device versus frequency.
- The derivative of the device's phase characteristic with respect to frequency.

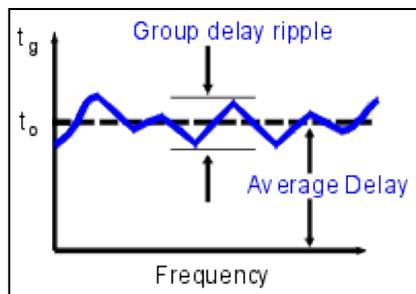
Refer to the graphic below for the following discussion:



The phase characteristic of a device typically consists of both linear and higher order (deviations from linear) phase-shift components.

Linear phase-shift component:	Higher-order phase-shift component:
Represents average signal transit time.	Represents variations in transit time for different frequencies.
Attributed to electrical length of test device.	Source of signal distortion.

Refer to the graphic below for the following discussion:



In a group delay measurement:

- The linear phase shift component is converted to a constant value (representing the average delay).
- The higher order phase shift component is transformed into deviations from constant group delay (or group delay ripple).
- The deviations in group delay cause signal distortion, just as deviations from linear phase cause distortion.
- The measurement trace depicts the amount of time it takes for each frequency to travel through the device under test.

Refer to the following equation for this discussion on how the PNA computes group delay:

$$\text{Group Delay} = t_g = \frac{-d\phi}{d\omega}$$

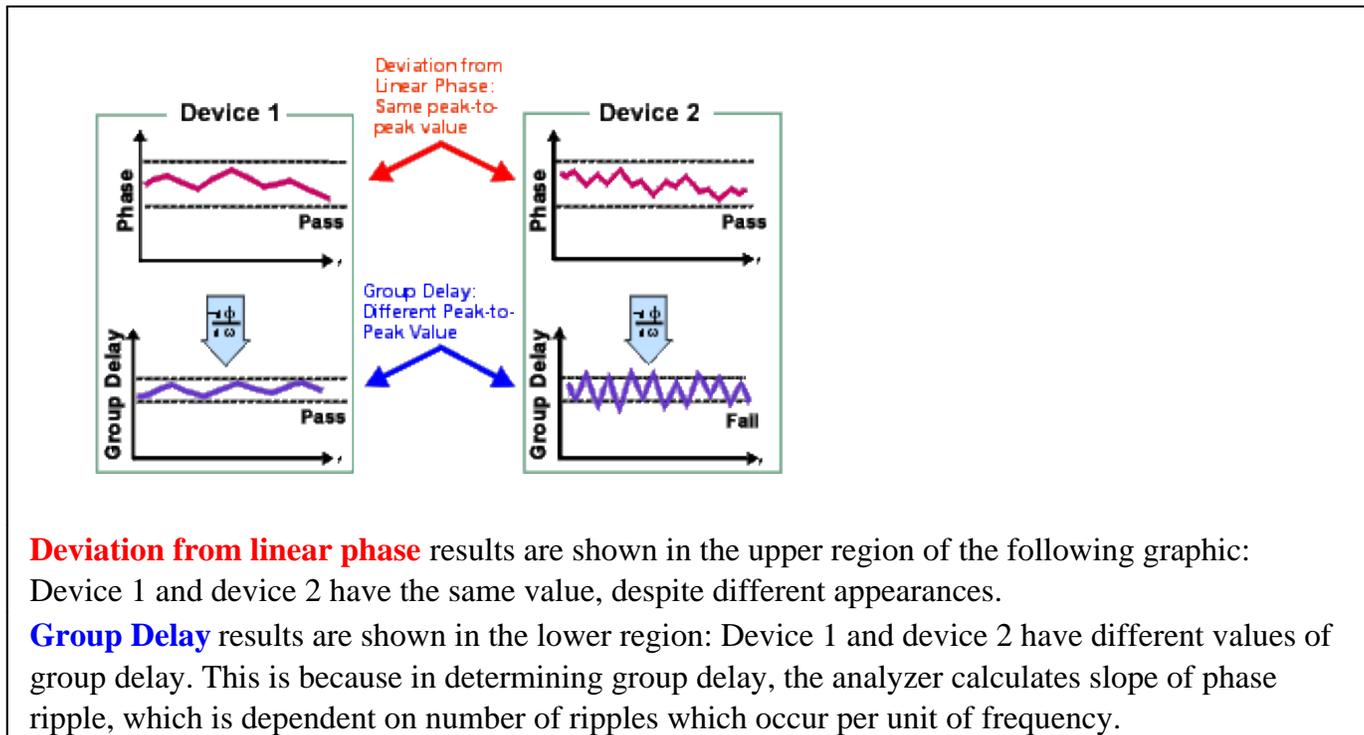
$$= \frac{-1}{360^\circ} \cdot \frac{d\Theta}{df}$$

ϕ in Radians
 ω in Radians/Sec
 Θ in Degrees
 f in Hz ($\omega = 2 \pi f$)

- Phase data is used to find the phase change ($-d\phi$).
- A specified frequency aperture is used to find the frequency change ($d\omega$).
- Using the two values above, an approximation is calculated for the rate of change of phase with frequency.
- This approximation represents group delay in seconds (assuming linear phase change over the specified frequency aperture).

Group Delay versus Deviation from Linear Phase

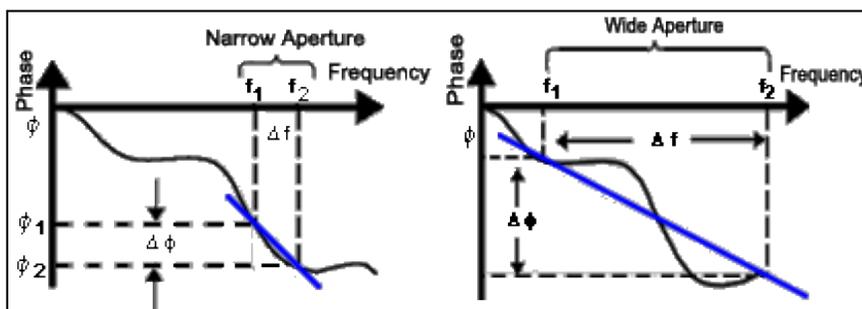
Group delay is often a more accurate indication of phase distortion than [Deviation from Linear Phase](#).



What Is Aperture?

During a group delay measurement, the PNA measures the phase at two closely spaced frequencies and then computes the phase slope. The frequency interval (frequency delta) between the two phase measurement points is called the aperture. Changing the aperture can result in different values of group delay. The computed slope ($-\Delta\phi / \Delta f$) varies as the aperture is increased. This is why when you are comparing group delay data, you must know the aperture that was used to make the measurements.

Refer to the graphic below for the following discussion:



Narrow aperture:	Wide aperture:
Provides more detail in phase linearity.	Provides less detail in phase linearity because some phase response averaged-out or not measured.
Makes measurement susceptible to noise (smaller signal-to-noise ratio) and PNA phase detector resolution.	Makes measurement less susceptible to noise (larger signal-to-noise ratio).

Group delay measurements can be made using the following [sweep types](#):

- Linear frequency
- List frequency sweep segment - The group delay aperture varies depending on the frequency spacing and point density. Therefore the aperture is not constant in segment sweep. In segment sweep, extra frequency points can be defined to ensure the desired aperture.

How to set Group Delay Aperture

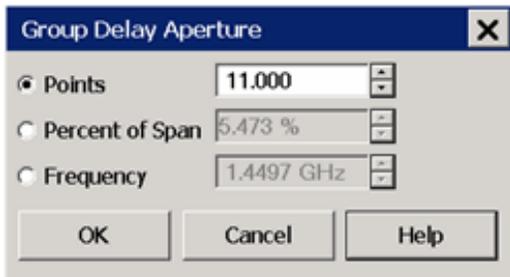
Using front-panel HARDKEY [softkey] buttons

1. Press **Avg**
2. then **[More]**
3. then **[Group Delay Aperture]**

PNA Menu using a mouse

1. Click **Response**
2. then **Avg**
3. then **Group Delay Aperture**

Programming Commands



Group Delay Aperture dialog box help

Although the Group Delay Aperture is defined as the difference in frequency between two data points (see [What Is Aperture?](#)), the group delay calculation can be averaged over many adjacent data points, similar to the PNA smoothing feature. The number of adjacent data points can be set using any of the following methods:

Note: You can change the default Group Delay Aperture to two points using a PNA Preference. [Learn how.](#)

Points Number of adjacent data points to average. Default setting is 11 points. Choose a value between 2 and the current number of points in the channel.

Percent of Span The data points within this percentage of the current frequency span are averaged. Choose a value between (2 points / current number of points) and 100 percent. The span must contain at least two data points.

Frequency The data points within this frequency range are averaged. The frequency range must contain at least two data points.

When the frequency span or number of points is reduced so that the current Group Delay Aperture is NOT attainable, the Aperture is adjusted to the new frequency span or number of points.

OK Applies setting changes and closes the dialog box.

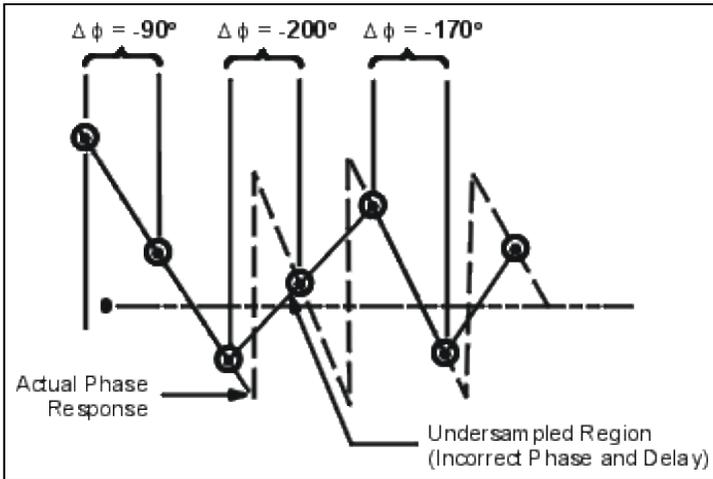
Cancel Closes the dialog. Setting changes are NOT applied.

Accuracy Considerations

It is important to keep the phase difference between two adjacent measurement points less than 180° (see the following graphic). Otherwise, incorrect phase and delay information may result. Undersampling may occur when measuring devices with long electrical length. You can verify that the phase difference measured between two adjacent points is less than 180° by adjusting the following settings until the measurement trace no longer changes:

- Increase the number of points
- Narrow the frequency span

Electrical delay may also be used to compensate for this effect.

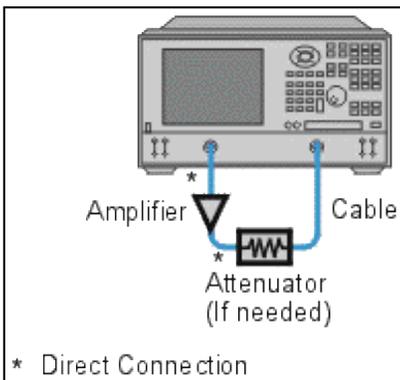


The frequency response is the dominant error in a group delay test setup. Performing a thru-response measurement calibration significantly reduces this error. For greater accuracy, perform a 2-port measurement calibration.

Particularly for an amplifier, the response may vary differently at various temperatures. The tests should be done when the amplifier is at the desired operating temperature.

How to Measure Group Delay

1. Preset the analyzer.
2. If your DUT is an amplifier, it may be necessary to adjust the PNA source power:
 - o Set the source power to be in the linear region of the amplifier's output response, typically 10 dB below the 1 dB compression point.
 - o If needed, use an external attenuator so the amplifier output power will be sufficiently attenuated to avoid causing receiver compression or damage to the PNA port 2.
3. Connect the DUT as shown in the following graphic.



4. Select an S21 measurement.

5. Select the settings for your DUT:
 - o frequency range
 - o number of measurement points.
 - o format: delay
 - o scale: autoscale
 6. Remove the DUT and perform a measurement calibration.
 7. Reconnect the DUT.
 8. Scale the displayed measurement for optimum viewing.
 9. Use the Group Delay Aperture setting to increase the aperture, reducing noise on the trace while maintaining meaningful detail.
 10. Use the markers to measure group delay (expressed in seconds) at a particular frequency of interest.
 11. Print the data or save it to a disk.
-

Last Modified:

23-Feb-2010	Added new dialog and text
4-Jan-2010	Fixed default aperture formula

High-Gain Amplifier Measurements

When measuring High-Gain Amplifiers, errors in measuring any of the S-parameters during calibration can result in error in the S21 measurement. This is because all the S-parameters are used in the error correction math.

A particular problem occurs with high gain amplifiers because the source power is set very low. Thus, when making reverse measurements (S22, S12) the signal-to-noise is poor and the raw measurements can be dominated by noise. This noise in the raw measurements will result in a noisy trace appearing for corrected S21 or S11.

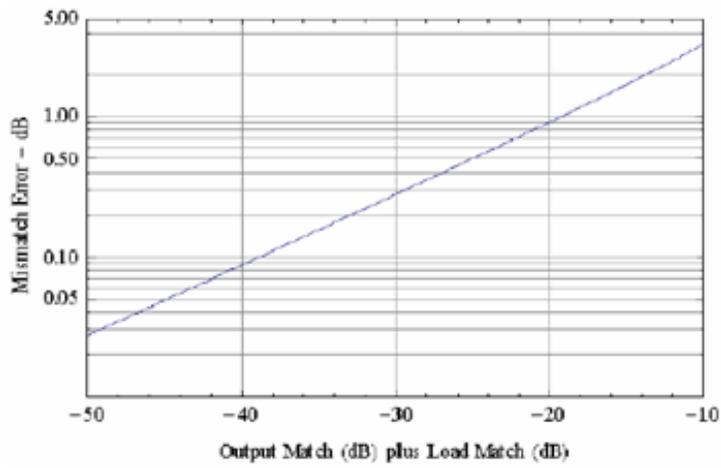
If you are using a large attenuator on port 2 (which improves output match), perform an Enhanced Response Calibration as follows. This corrects for the same errors as the full 2-port correction EXCEPT the interaction between the raw load match and the DUT output match.

1. There is NO need to Uncouple the port powers.
2. Set port powers to an acceptable level. Do NOT overpower the PNA test port.
3. Perform Enhanced Response Cal. [Learn how.](#) (Does not measure or correct for the S12 or S22 / PNA port match).

If you want to do a full correction (for example, when your amplifier output match is poor so the Enhanced Response Cal above is not adequate), then...

1. Uncouple the port powers. [Learn how.](#)
2. Set input (port 1) power to approximately the output power of the amplifier up to 0 dBm
3. Set reverse (port 2) power to the same power (for measuring isolation and S22)
4. Perform a Full 2-port Cal.
5. Re-set the input power (port 1) to a lower power level appropriate for driving the amplifier.

Additional Error due to Mismatch of DUT Output Match and Raw Load Match

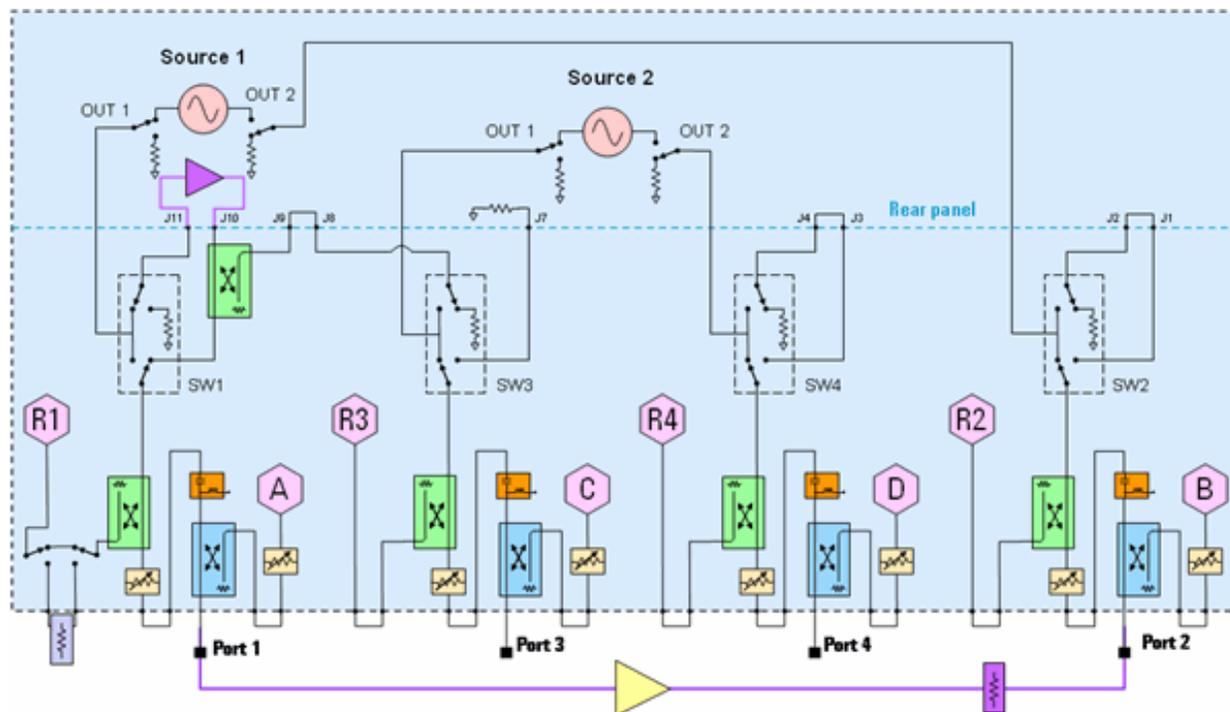


Last Modified:

23-Sep-2009 MX New topic

High-Power Amplifier Measurements with the PNA-X

The following is a block diagram of the [PNA-X Opt 423](#). The configuration displayed here is used to make high power amplifier measurements using a preamplifier at the rear panel. The preamplifier can then be switched (SW1) as needed using the [RF Configurator](#).



Legend

Color	Component	Damage Level
Green	Bridges	+33 dBm
Blue	Couplers	+43 dBm
Orange	Bias-tees	+30 dBm
Purple	User-supplied pre-amp and high-power attenuator	N/A

Notes

At J11 (rear-panel), max power is 4 dB to 11 dB higher than Source 1 Out at front panel jumper due to loss of the coupler thru arms, bias-tees, and cables.

At J10 max power +33 dBm, which is the damage level of the bridge. With +30 dBm into J10, there will be about +15 dBm at R1, assuming 15 dB coupling factor for the R1 bridge. +15 dBm is the damage level of that receiver. Therefore, it may be necessary to add attenuation in place of the R1 loop, not only to protect the receiver, but to bring it out of compression. The 0.1 dB compression level spec

for the R1 receiver is between -3 and -18 dBm, depending on the frequency and option configuration.

At Test Port 2 (DUT output): With the bias-tees (orange), only +30 dBm is allowed into the test port. With Opt H85 (bias-tees removed), +43 dBm is allowed. Add appropriate attenuation to not damage other components.

About the PNA-X Option H85

A modified version of the standard PNA-X, the option H85 is designed to permit insertion of high power amplifiers and other signal conditioning equipment to allow high power network measurements at RF levels up to 20 Watts (+43 dBm) from 10 MHz to 26.5 GHz. The Option H85 modification supplies extended power range attenuators without bias tees. This is similar to the PNA-X -219 (add extended power range and bias-tees to 2-Port analyzer) or PNA-X -419 (add extended power range and bias-tees to 4-Port analyzer) but deletes the bias tees from the test set.

See Also

- [High-Gain Amplifier Measurements](#)
- [N5242-H85 Manual](#)
- [RF Path Configurator](#)
- [IF Path Configurator](#)
- [High-power measurements using the PNA \(5989-1349EN\)](#) Application Note 1408-10 at Agilent.com.

Last Modified:

6-Apr-2009	Replaced N5242A with PNA-X
14-Aug-2008	Added link to H85 manual
10-May-2007	MX New topic

Phase Measurements

Knowledge of both magnitude and phase characteristics is needed for successful higher-level component integration.

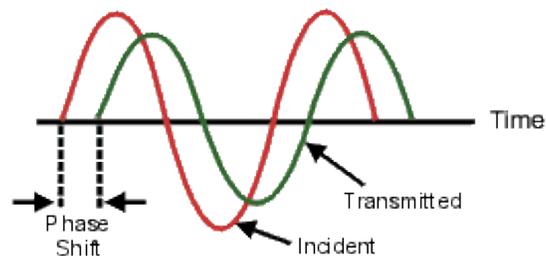
- [What are Phase Measurements?](#)
- [Why Measure Phase?](#)
- [Using the Analyzer's Phase Format](#)
- [Types of Phase Measurements](#)

See other Tutorials

What are Phase Measurements?

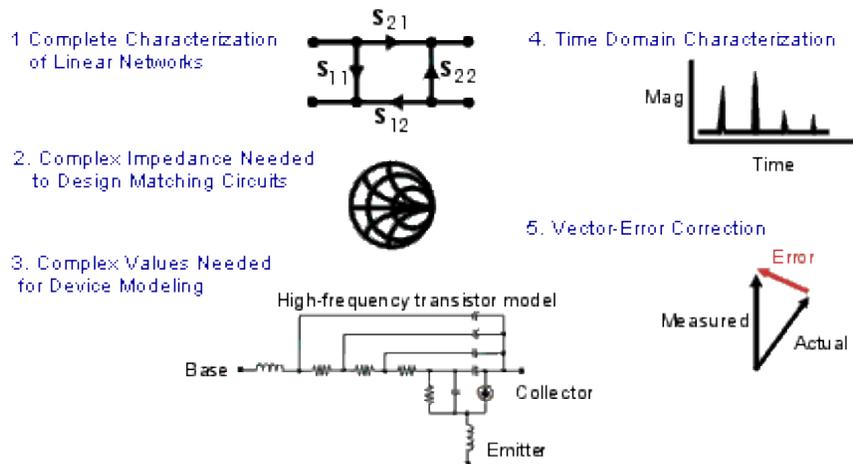
Phase measurements are made using S-parameters, just like amplitude measurements. A phase measurement is a relative (ratio) measurement and not an absolute measurement. Phase measurements compare the phase of the signal going into a device (the incident signal) to the phase of the device's response signal. The response signal can be either reflected or transmitted. Assuming an accurate **calibration** has been performed, the difference in phase between the two signals (known as phase shift) is a result of the electrical characteristics of the device under test.

The following graphic shows the phase shift (in time or degrees) between an incident signal and a transmitted signal (as might be seen on an oscilloscope display).



Why Measure Phase?

Measuring **phase** is a critical element of network analysis. The following graphic lists five reasons for measuring both magnitude and phase.



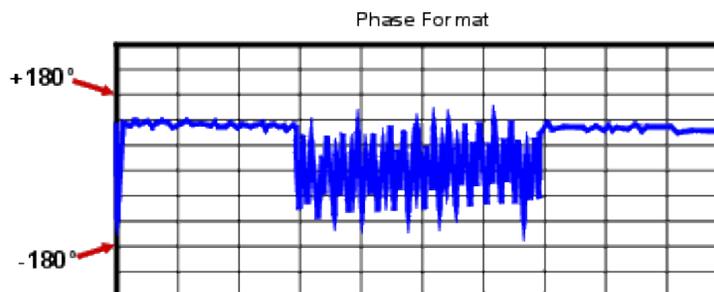
When used in communications systems to pass signals, components or circuits must not cause excessive signal distortion. This distortion can be:

- Linear, where flat magnitude and linear phase shift versus frequency is not maintained over the **bandwidth** of interest.
- Nonlinear, such as AM-to-PM conversion.

It is important to measure how reflective a component or circuit is, to ensure that it transmits or absorbs energy efficiently. Measuring the complex **impedance** of an antenna is a good example.

Using the Analyzer's Phase Format

The analyzer's phase format displays a phase-versus-frequency or phase-versus-power measurement. The analyzer does not display more than ± 180 degrees phase difference between the reference and test signals. As the phase value varies between $+180$ degrees and -180 degrees, the analyzer display creates the sawtooth pattern as shown in the following graphic.



The sawtooth pattern does not always reach $+180$ degrees and -180 degrees. This is because the measurement is made at discrete frequencies, and the data point at $+180$ degrees and -180 degrees may not be measured for the selected sweep.

Types of Phase Measurements

[Complex impedance](#) data is information such as resistance, reactance, phase, and magnitude that can be determined from an S11 or S22 measurement. Complex impedance data can be viewed using either the Smith Chart format or the Polar format.

[AM-to-PM conversion](#) is a measure of the amount of undesired phase deviation (PM) that is caused by amplitude variations (AM) of the system. AM-to-PM conversion is usually defined as the change in output phase for a 1-dB increment in the input power to an amplifier (i.e. at the 1 dB gain compression point). This is expressed in degrees-per-dB ($^{\circ}/\text{dB}$).

[Deviation from linear phase](#) is a measure of phase distortion caused by a device. Ideally, the phase shift through a device is a linear function of frequency. The amount of variation from this theoretical phase shift is known as its deviation from linear phase (also called phase linearity).

[Group delay](#) is another way to look at phase distortion caused by a device. Group delay is a measure of transit time through a device at a particular frequency. The analyzer computes group delay from the derivative of the measured phase response.

Deviation from Linear Phase Versus Group Delay

Although deviation from linear phase and group delay are similar measurements, they each have their purpose.

The following are the advantages of deviation from linear phase measurements:

- Less noisy than group delay.
- Able to characterize devices that pass phase modulated signals, and show units of phase rather than units of seconds.

The following are the advantages of group delay measurements:

- More easily interpreted indication of phase distortion than deviation from linear phase.
- Able to most accurately characterize a device under test. This is because in determining group delay, the analyzer calculates the slope of the phase ripple, which is dependent on the number of ripples which occur per unit of frequency. Comparing two phase responses with equal peak-to-peak phase ripple, the response with the larger phase slope results in:
 - More group delay variation.
 - More signal distortion.

See also [Comparing the PNA Delay Functions](#).

Reverse Isolation

Reverse isolation is a measure of amplifier reverse transmission response- from output to input.

- [What is Reverse Isolation](#)
- [Why Measure Reverse Isolation?](#)
- [Accuracy Considerations](#)
- [How to Measure Reverse Isolation](#)

[See other Tutorials](#)

What is Reverse Isolation?

Reverse isolation is a measure of how well a signal applied to the device output is "isolated" from its input.

The measurement of reverse isolation is similar to that of forward gain, except:

- The stimulus signal is applied to the amplifier's output port.
- The response is measured at the amplifier's input port.

The equivalent S-parameter is S_{12} .

Why Measure Reverse Isolation?

An ideal amplifier would have infinite reverse isolation-no signal would be transmitted from the output back to the input. However, reflected signals can pass through the amplifier in the reverse direction. This unwanted reverse transmission can cause the reflected signals to interfere with the desired fundamental signal flowing in the forward direction. Therefore, reverse isolation is important to quantify.

Accuracy Considerations

Since amplifiers often exhibit high loss in the reverse direction, generally there is no need for any **attenuation** that may have been used to protect the port 2 receiver during forward transmission measurements. Removing the attenuation will:

- Increase the **dynamic range**, resulting in improved measurement accuracy.
- Require a new **calibration** for maximum accuracy.

The RF source power can be increased to provide more dynamic range and accuracy.

Note: With the attenuation removed and the RF source power increased, a forward sweep could damage the analyzer's port 2 receiver. Do not perform a forward sweep or use 2-port calibration unless the forward power is set low enough to avoid causing port 2 receiver compression or damage.

If the **isolation** of the amplifier under test is very large, the transmitted signal level may be near the **noise floor** or **crosstalk** level of the receiver. To lower the noise floor:

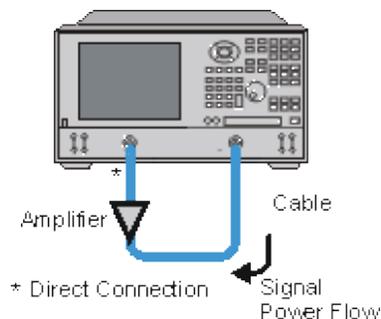
- Use or increase measurement averages.
- Reduce the IF **bandwidth** of the analyzer.

Note: Reducing IF bandwidth or using **averaging** improves measurement dynamic range and accuracy, at the expense of reduced measurement speed.

- When crosstalk levels affect the measurement accuracy, reduce the crosstalk error term by performing a response and isolation calibration. When performing the isolation part of the calibration it is important to use the same average factor and IF bandwidth during the calibration and measurement.
- The **frequency response** of the test setup is the dominant error in a reverse isolation measurement. Performing a thru-response measurement calibration significantly reduces this error. This calibration can be done as part of the response and isolation calibration.
- The amplifier may respond very differently at various temperatures. The tests should be done when the amplifier is at the desired operating temperature.

How to Measure Reverse Isolation

1. Connect the amplifier as shown in the following graphic.



2. Preset the analyzer.

3. Select an S12 measurement.
4. Select the settings for your amplifier under test.
5. Remove the amplifier and perform a thru-response **calibration** or a response and **isolation** calibration.
6. Scale the displayed measurement for optimum viewing and use a marker to measure the reverse isolation at a desired frequency.
7. Print or save the data to a disk.

Reflection Measurements

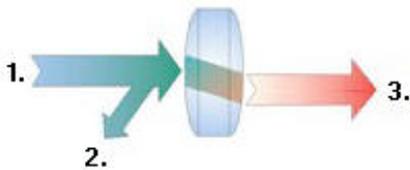
Reflection measurements are an important part of network analysis.

- [What are Reflection Measurements?](#)
- [Why Make Reflection Measurements?](#)
- [Expressing Reflected Waves](#)
 - [Return Loss](#)
 - [VSWR](#)
 - [Reflection Coefficient](#)
 - [Impedance](#)
 - [Summary of Expressions](#)

[See other Tutorials](#)

What are Reflection Measurements?

To understand reflection measurements, it is helpful to think of traveling waves along a transmission line in terms of a lightwave analogy. We can imagine incident light striking some optical component like a clear lens. Some of the light is reflected off the surface of the lens, but most of the light continues on through the lens. If the lens had mirrored surfaces, then most of the light would be reflected and little or none would be transmitted.



1. Incident 2. Reflected 3. Transmitted

With RF energy, reflections occur when the impedance of two mated devices are not the same. A reflection measurement is the ratio of the reflected signal to the incident signal. Network analyzers measure the incident wave with the R (for reference) channel and the reflected wave with the A

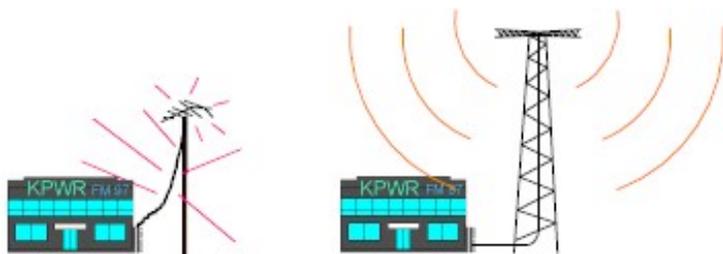
channel. Therefore, reflection is often shown as the ratio of A over R (A/R). We can completely quantify the reflection characteristics of our device under test (DUT) with the amplitude and phase information available at both the A and R channel. In S-parameter terminology, S11 is a reflection measurement of port1 of the device (the input port); S22 is a reflection measurement of the port 2 (the output port)

Why Make Reflection Measurements?

One reason we make reflection measurements to assure efficient transfer of RF power. We do this because:

1. RF energy is not cheap. When energy is reflected, that means less energy is transmitted to where it is intended to go.
2. If the reflected energy is large, it can damage components, like amplifiers.

For example, in the following graphic, the radio station on the left is not operating at peak efficiency. The amplifier impedance is not the same as the transmission line, and the transmission line impedance is not the same as the antenna. Both of these conditions cause high reflected power. This condition results in less transmitted power, and the high reflected power could damage the amplifier.



The radio station on the right installed properly "matched" transmission line and antenna. Very little of the transmitted signal is reflected, resulting in increased broadcast power, more listeners, more advertising revenue, and more profit. The amplifier, transmission, and antenna all need to be measured to ensure that reflected power is minimized.

Expressing Reflected Waves

After making a reflection measurement, the reflection data can be expressed in a number of ways, depending on what you are trying to learn. The various expressions are all calculated by the analyzer from the same reflection measurement data. Each method of expressing reflection data can be graphically displayed in one or more formats. For more information, see display formats.

Return Loss

The easiest way to convey reflection data is return loss. Return loss is expressed in dB, and is a scalar (amplitude only) quantity. Return loss can be thought of as the absolute value or dB that the reflected

signal is below the incident signal. Return loss varies between infinity for a perfect impedance match and 0 dB for an open or short circuit, or a lossless reactance. For example, using the log magnitude format on the analyzer, the measured reflection value on the screen may be -18dB. The minus sign is ignored when expressing return loss, so the component is said to have 18dB of return loss.

VSWR

Two waves traveling in opposite directions on the same transmission line cause a "standing wave". This condition can be measured in terms of the voltage standing wave ratio (VSWR or SWR for short). VSWR is defined as the maximum reflected voltage over the minimum reflected voltage at a given frequency. VSWR is a scalar (amplitude only) quantity. VSWR varies between one for a perfect match, and infinity for an open or short circuit or lossless reactance.

Reflection Coefficient

Another way of expressing reflection measurements is reflection coefficient gamma (Γ). Gamma includes both magnitude and phase.

The magnitude portion of gamma is called rho (ρ). Reflection coefficient is the ratio of the reflected signal voltage to the incident signal voltage. The range of possible values for ρ is between zero and one. A transmission line terminated in its characteristic impedance will have all energy transferred to the load; zero energy will be reflected and $\rho = 0$. When a transmission line terminated in a short or open circuit, all energy is reflected and $\rho = 1$. The value of rho is unitless.

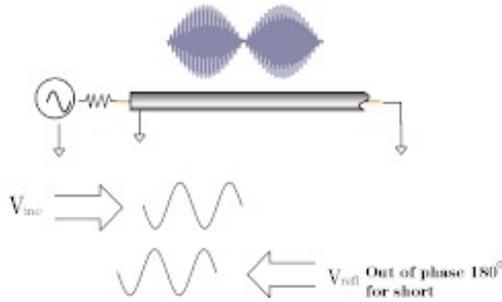
Now for the phase information. At high frequencies, where the wavelength of the signal is smaller than the length of conductors, reflections are best thought of as waves moving in the opposite direction of the incident waves. The incident and reflected waves combine to produce a single "standing" wave with voltage that varies with position along the transmission line.

When a transmission line is terminated in its characteristic impedance (Z_0) there is no reflected signal. All of the incident signal is transferred to the load, as shown in the following graphic. There is energy flowing in one direction along the transmission line.

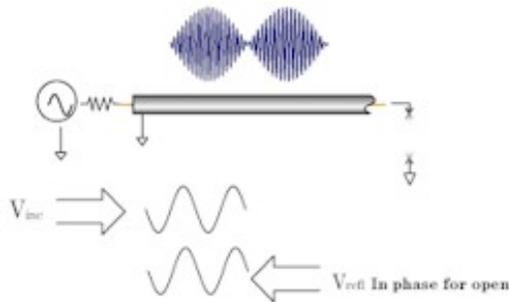


When a transmission line is terminated in a short circuit termination, all of the energy is reflected back to the source. The reflected wave is equal in magnitude to the incident wave ($\rho = 1$). The voltage across any short circuit is zero volts. Therefore, the voltage of the reflected wave will be 180 degrees out of

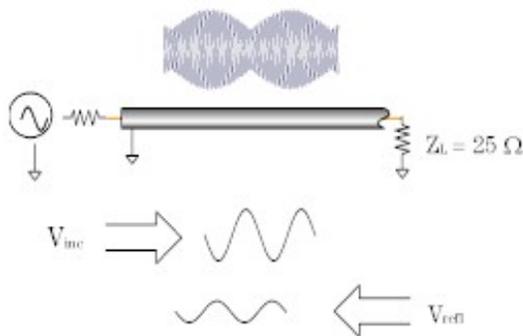
phase with the incident wave, canceling the voltage at the load.



When a transmission line is terminated in an open circuit termination, all of the energy is reflected back to the source. The reflected wave is equal in magnitude to the incident wave ($\rho = 1$). However, no current can flow in an open circuit. Therefore, the voltage of the reflected wave will be in phase with the voltage of the incident wave.



When a transmission line is terminated in a 25 ohm resistor, some but not all of the incident energy will be absorbed, and some will be reflected back towards the source. The reflected wave will have an amplitude 1/3 that of the incident wave and the voltage of the two waves will be out of phase by 180 degrees at the load. The phase relationship will change as a function of distance along the transmission line from the load. The valleys of the standing wave pattern will no longer go to zero, and the peaks will be less than that of the open / short circuit.

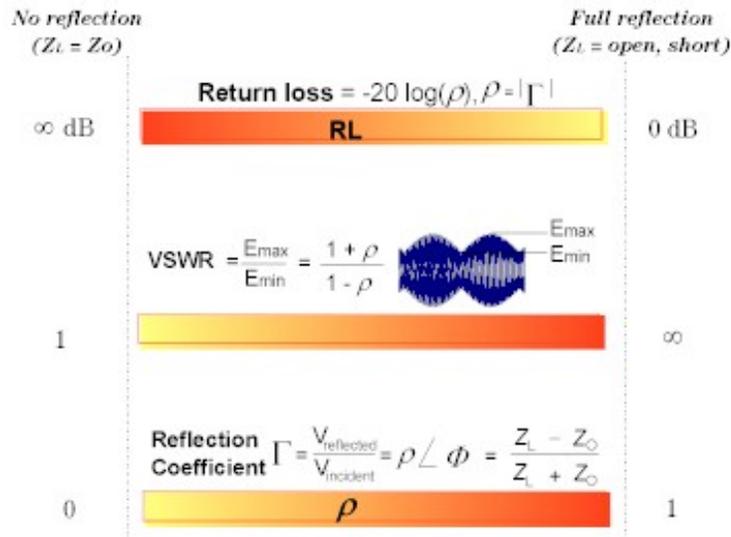


For more information, see [Phase Measurements](#).

Impedance

Impedance is another way of expressing reflection data. For more information on Impedance, see [Smith Charts](#).

Summary of the Expressions of Reflection Measurements:



Specifications - Select the PNA Model

Please select the PNA model of the specifications you would like to see.

Note: Beginning with PNA Rev. A.09.33, PNA specs are no longer embedded in PNA Help.

An internet connection is required to view ALL specifications documents.

Doc Number	Model
PNA-X Models	
N5242-90007	N5241A (13.5 GHz)
	N5242A (26.5 GHz)
N5245-90008	N5244A (43.5 GHz)
	N5245A (50 GHz)
N5247-90002	N5247A (67 GHz)
N523x Models	
N5235-90003	N5234A (10 MHz to 43.5 GHz - 2-port ONLY)
	N5235A (10 MHz to 50.0 GHz - 2-port ONLY)
N5235-90004	N5239A (300 kHz to 8.5 GHz - 2-port ONLY)
	N5231A (300 kHz to 13.5 GHz - 2-port / 4-port)
	N5232A (300 kHz to 20.0 GHz - 2-port / 4-port)
N522xA Models	
N5221-90001	N5221A (13.5 GHz)
	N5222A (26.5 GHz)
N5224-90001	N5224A (43.5 GHz)
	N5225A (50 GHz)
N5227-90002	N5227A (67 GHz)
N522xA "Metrology" Models	
Opt 210/410	

N5221-90002	N5221A (13.5 GHz)
	N5222A (26.5 GHz)
N5224-90002	N5224A (43.5 GHz)
	N5225A (50 GHz)
N5227-90003	N5227A (67 GHz)
Others	
N5264-90003	N5264A (Measurement Receiver)
N5245-90016	N5244A or N5245A with Option H29 (Noise Figure to 26.5 GHz)
E8361-90007	E8361A/C (67 GHz)
E8364-90031	E8362A/B/C (20 GHz)
	E8363A/B/C (40 GHz)
	E8364A/B/C (50 GHz)
N5230-90016	N5230A/C 2-Port (6, 13, 20, 40, 50 GHz)
N5230-90020	N5230A/C 4-Port (13.5 and 20 GHz)

See the [equations that are used to generate uncertainty curves](#).

Block Diagrams for the following models are available in PNAHelp:

- [PNA-X Models with Opt 224 and 423](#)
- [N522xA Models with Opt 219 and 419](#)

The following are specifications for discontinued models:

- [E8356A, E8357A, E8358A](#)
- [E8801A, E8802A, E8803A](#)
- [N3381A, N3382A, N3383A](#)
- [E8362A, E8363A, E8364A](#)

Glossary

[A](#) [B](#) [C](#) [D](#) [E](#) [F](#) [G](#) [H](#) [I](#) [J](#) [K](#) [L](#) [M](#)
[N](#) [O](#) [P](#) [Q](#) [R](#) [S](#) [T](#) [U](#) [V](#) [W](#) [X](#) [Y](#) [Z](#)

12-Term Error Correction See [Error Correction, 12-Term](#).

1-Port Device A device with a single connector or path to the device's circuitry. Examples include an oscillator and a load.

2-Port Calibration, Full See [Error Correction, 12-Term](#).

2-Port Device A device with two connectors or other paths to the device's circuitry. Examples include filters, SAW devices, attenuators, matching pads, and amplifiers.

3-Term Error Correction See [Error Correction, 3-Term](#).

A

Active Channel The highlighted channel affected by front panel functions.

Active Function Readout The area of a display screen where the active function and its state are displayed. The active function is the one that was completed by the last key selection or remote programming command.

Active Marker The marker on a trace that can be repositioned either by front panel controls or by programming commands.

Active Trace A trace that is being swept (updated) with incoming signal information.

ADC Analog to Digital Converter

Address The identification (represented by a name, label, or number) for a register, location in storage, or any other data source or destination. Examples are the location of a station in a communications network, or a device on the GP-IB.

ADM Add-Drop Multiplexer

Admittance (Y) The inverse of an impedance (i.e. the ratio of current to voltage). Complex admittances take the form $Y = G + jB(t)$.

ALC Automatic Level Control. See [Automatic Gain Control](#).

AM Amplitude Modulation

AM Group Delay A technique for the measurement of group delay through a device which utilizes an amplitude modulated (AM) source. Note: The actual delay of the modulation envelope is measured directly with an external scalar detector. Devices that distort the amplitude of a signal cannot be measured. These include amplifiers with automatic gain control (AGC) and devices subject to saturation or power limiting.

Amplitude Modulation The process, or result of the process, of varying the amplitude of a carrier signal. The resulting modulated carrier contains information that can be recovered by demodulation. See also [Modulation](#).

Analog The general class of devices or circuits in which the output varies as a continuous function of the input.

Annotation The labeling of specific information on the display (such as frequency or power).

ANSI American National Standards Institute: A national membership organization (open to manufacturers, organizations, users, and communications carriers) that approves standards, accredits standards development groups and certificate programs, and represents and coordinates US interests in non-treaty and non-government standards bodies.

Aperture The frequency span of the network analyzer used for calculating group delay. The narrower the aperture, the finer the resolution of the group delay variations, but noise is reduced by increasing the aperture.

Array A set of numbers or characters that represents any given function.

ASCII American Standard Code for Information Interchange

Attenuation Denotes a reduction in signal amplitude. The difference between transmitted and received power due to loss through equipment, lines, or other transmission devices; usually expressed in decibels.

Attenuator An RF or microwave device used to reduce the power level of a signal by precise, incremental amounts over its entire frequency range.

Automatic Calibration System AutoCal: Feature offered on Rohde&Schwarz network analyzers.

Automatic Gain Control (AGC) A circuit used in amplifiers and other active devices to keep its RF power level constant as other parameters change, such as frequency. Synonym: Automatic Leveling Control (ALC)

Autoscale An analyzer feature that evaluates waveforms and adjusts controls to stable and enhance the display.

AUX Auxiliary; refers to rear-panel input connector.

Averaging A noise reduction technique that computes each data point based on consecutive sweeps and weighted by a user-specified averaging factor. Each new sweep is averaged into the trace until the total number of sweeps is equal to the averaging factor.

B

B/R The ratio of data sampled at B to the data sampled at R.

Band Pass A range of frequencies that are passed through a device, such as a filter. Frequencies not within the band pass are limited or attenuated. See also [Cutoff Frequency](#).

Bandwidth (BW) The difference between the frequencies of a continuous frequency band within which performance of a device falls within specifications.

Bandwidth Limit The condition prevailing when the system bandwidth is exceeded and signal distortion occurs beyond specifications.

Bandwidth Selectivity A measure of a filter's ability to resolve signals unequal in amplitude. It is the ratio of the 60 dB bandwidth to the 3 dB bandwidth for a given resolution filter (IF). Bandwidth selectivity tells us how steep the filter skirts are. Bandwidth selectivity is sometimes called shape factor.

Binary A method of representing numbers in a scale of two (on or off, high-level or low-level, one or zero). A compact, fast format used to transfer information to and from the analyzer.

BMP Bit-Mapped

Brightness See [Color Brightness](#).

Broadband Device A device that operates over a very wide frequency range and exhibits only small variations in response over that range.

Buffer A storage device used when transmitting information to compensate for a difference in the rate of flow of information between two devices.

Burst Carrier A carrier that is periodically turned off and on. A burst carrier may or may not be modulated.

BUS Basic Utility System

Bus One or more conductors used as a path to deliver transmitted information from any of several sources to any of several destinations.

BW Bandwidth

Byte Eight bits of data representing one character processed as a unit.

C

CAD Computer Aided Design

CAE Computer Aided Engineering

Calibration In HP instrumentation, the process of periodically (usually annually) verifying an instrument is performing to specifications. A calibration certificate is awarded after verification.

In network analyzers, the process of removing systematic errors from measurements. See [Error Correction](#).

Calibration Kit Hardware and software required to perform error correction on a network analyzer for a specific measurement and/or test set.

Calibration, 2-Port See [Error Correction, 12-Term](#).

Calibration, Blackburn Calibrations of transmission path with corrected source match involving 15 calibration terms. Synonym: 15-term error correction

Calibration, Frequency Response The simplest error correction procedure to perform, but only corrects for a few of the twelve possible systematic error terms. Frequency response corrections can be made for reflection measurements, transmission measurements, and isolation measurements.

Calibration, Interpolation A user selectable network analyzer feature that calculates (interpolates) new error correction terms from existing terms when there is a change in network analyzer parameters, such as IF bandwidth, power, or sweep time. The resulting error correction is not as accurate as completing a full 2-port calibration.

Calibration, Port Extension See [Port Extension](#).

Calibration, Reference Plane See [Reference Plane](#).

Calibration, Set Z Sets the system impedance, usually 50 or 75 ohms.

Calibration, SOLT A calibration using four known standards: Short-Open-Load-Through. Also known as a full two-port calibration and 12-term error correction. See also [Error Correction](#).

Calibration, TRL and LRM A calibration used in environments where the DUT cannot be connected directly to the network analyzer ports, (MMIC, microstrip, beam-lead diodes etc.). Thru-Reflect-Line (TRL) and M (Match) standards are fabricated and used because known high-quality standards are not

readily available. The requirements for characterizing these standards are less stringent, but the calibration is not as accurate as the traditional full two-port calibration using S-O-L-T standards. The terms are used interchangeably (TRL, LRL, LRM etc.) but they all refer to the same basic calibration method.

Characteristic Impedance The impedance looking into the end of an infinitely long lossless transmission line.

Color Brightness A measure of the intensity (brightness) of a color.

Command A set of instructions that are translated into instrument actions. The actions are usually made up of individual steps that together can execute an operation.

Continuous Sweep Mode The analyzer condition where traces are automatically updated each time trigger conditions are met.

Controller A device capable of specifying the talker and listeners for an information transfer. An external computer connected to an instrument to control its operation.

Corrected Measurements made after performing error correction.

Coupler See [Directional Coupler](#).

CPU Central Processing Unit

Crosstalk The occurrence of a signal at one port of a device being affected by a signal in any other path. Isolation is the measurement of crosstalk.

Cursor An electronically generated pointer that moves across the display to manipulate controls.

Cutoff Frequency In filters, the frequency at which attenuation is 3dB below the band pass signal level, known as the 3dB points.

CW Continuous wave: A single frequency (rather than a swept frequency).

D

DAC Digital to Analog Converter

dB Decibel: a relative unit of measure. The ratio in dB is given by: $10 \log_{10} (P_1/P_2)$ where P_1 and P_2 are the measured powers. The dB is preferred instead of arithmetic ratios or percentages because when components are connected in series, their effect on power, expressed in dB, may be arithmetically added and subtracted. For example, if a 3dB attenuator is connected to a 10dB amplifier, the net gain of the two components is $(-3\text{dB} + 10\text{dB} = +7\text{dB})$.

dBm Absolute unit of measure in decibels: $0\text{dBm} = 1\text{ mW}$. The conventions of the dB (adding and subtracting) continue to apply.

DBMS Database Management System

DC Direct Current

Default A known set of conditions used in the absence of user-defined conditions.

Delay See [Group Delay](#).

Demodulation The process of recovering from a modulated carrier, information in the form of a signal having essentially the same characteristics as the original modulating signal. Recovery of the modulating signal accomplished by signal detection.

Detection The process of demodulating signal carriers. There are two basic ways of providing signal detection in network analyzers: Diode detectors (used in broadband applications) and heterodyning, (used in narrowband applications).

Detector, Diode A device used to convert a RF signal to a proportional DC level. If the signal is amplitude modulated, the diode strips the RF carrier signal from the modulation. Many sources used with scalar analyzers are amplitude modulated with a 27.778 kHz signal and then detected in the network analyzer. Phase information on the signal carrier is lost in diode detection.

Deviation from Linear Phase Linear phase refers to the nature of the phase shift of a signal through a device. The phase is linear if a plot of phase shift versus frequency is a straight line using linear scales. Deviation from linear phase causes signal distortion.

Digital Pertaining to the class of devices or circuits in which the output varies in discrete steps.

Digital Demodulation Describes a technique of extracting the information used to modulate a signal. Digital signal processing algorithms are used on the signal after it has been converted from an analog to a digital form (digitized).

Dimension To specify the size of an array. The number of array rows or columns.

Directivity In a 3-port directional coupler, the ratio of the power present at the auxiliary port when the signal is traveling in the forward direction to the power present at the auxiliary port when the same signal is traveling in the reverse direction.

Directional Coupler A 3-port device typically used for separately sampling the backward (reflected) wave in a transmission line.

Disk A circular, magnetic storage medium.

Display Noun: See [Screen](#).

Verb: To show annotation and measurement data on the display.

Display Detector Mode The manner in which analog, video information is processed prior to being digitized and stored in memory.

Display Dynamic Accuracy The amplitude uncertainty, usually in dB, over the display dynamic range.

Display Dynamic Range The amplitude range, in dB, over which the display dynamic accuracy applies.

Display Formats Graphical formats for displaying measurement data. These include single channel, overlay (multiple traces on one graticule), split (each trace on separate graticules).

Display Modes The ways in which measurement data can be presented graphically. On a network analyzer, the choices are Cartesian/rectilinear (XY plot with log or linear magnitude, phase, group delay, SWR, real and imaginary, and dBV, dBmV and dBuV), polar (magnitude and angle), magnitude and phase, and Smith chart. Not all display modes are available on all network analyzers. In addition, displays can present this information in various combinations of traces. Common modes are dual, (the ability to display more than one trace, usually over the same frequency range), and alternate, (the ability to display more than one trace, each with different frequency range and type).

Display Phase Dynamic Accuracy The phase measurement uncertainty, usually in degrees, for measurements whose units are in degrees.

Display Points The total number of measurement points made in a single measurement. The points can be in units of frequency, power, or time. The number of points often dictates measurement speed, resolution, and aperture.

Display Trace Noise, Magnitude The amplitude uncertainty of the trace, in dB, due to random noise in the test system.

Display Trace Noise, Phase The phase uncertainty of the trace, in degrees, due to random noise in the test system.

Display Type The type of display screen built into the analyzer. Data can be displayed as a raster drawing (a computer-like dot map) or as a vector drawing (lines drawn on the display). Color and display standard can also be specified as monochrome (single color), or color (two or more colors). The format standard may also be specified, such as VGA or SVGA, for IBM-compatible personal computers.

Distortion Deterioration of a signal's quality due to the nonlinear characteristics of a device or system transfer function. Distortion is measured as a combination of the changes in amplitude, frequency and phase of signal at the output of a device or system as compared to the signal at the input.

Drift The slow change in signal frequency.

DSP Digital Signal Processing

DUT Device Under Test

DVM Digital Volt Meter

Dynamic Range In a receiver, the range of signal levels, from minimum to maximum, that can be reliably measured simultaneously. Dynamic range allows small signals to be measured in the presence of large signals. Source power and receiver compression usually limits the maximum boundary to dynamic range. Receiver residual responses and noise floor usually limit the minimum power boundary.

E

ECal See [Electronic Calibration](#).

Electrical Delay A simulated variable length of lossless transmission line, added to or subtracted from a receiver input, to compensate for interconnecting cables. The firmware equivalent of mechanical or analog "line stretchers" in other network analyzers.

Electronic Calibration (ECal) A calibration system for electronic calibration of RF and microwave vector network analyzers. The electronic calibration system creates a twelve-term, two-port error model and then provides a confidence check of the calibration. The Ecal system consists of a repeatable, variable-impedance, solid-state calibration standard and a mainframe control unit which interfaces with the 8510, 8720 series, and the 8753 network analyzers or a USB module which interfaces with the PNA series network analyzers.

EMC Electro-Magnetic Compatibility

EMI Electro-Magnetic Interference: Unintentional interfering signals generated within or external to electronic equipment. Typical sources could be power-line transients, noise from switching-type power supplies and/or spurious radiation from oscillators. EMI is suppressed with power-line filtering, shielding, etc.

Engage To activate a function.

Enter The process of inputting information.

EPROM Electronically Programmable, Read-Only Memory

Error Correction In network analyzers, a process that removes or reduces systematic (repeatable) measurement errors by measuring known standards from a calibration kit. Synonym: measurement calibration

Error Correction, 3-Term Used to remove systematic measurement errors on a device with one port, such as a load.

Error Correction, 12-Term Correction for a two port device using six parameters:

Directivity

Source match

Load match

Reflection frequency response

Transmission frequency response

Isolation

To completely characterize a two-port device, these six parameters must be characterized in the forward and reverse directions, making a total of 12 terms. The user usually has the option of omitting isolation from the correction process. Synonym: Full two-port error correction

Error Correction, 1-Port Corrects a test set for port 1 or port 2 directivity, frequency response, and source match errors. The process requires three known standard terminations, for example, open, short, and load.

Error Message A message on a display that indicates an error condition. Missing or failed hardware, improper user operation, or other conditions that require additional attention can cause an error condition. Generally, the requested action or operation cannot be completed until the condition is resolved.

ESD Electro Static Discharge

Ethernet A network that adheres to the IEEE 802.3 Local Area Network standard.

Ethernet address A hexadecimal number which is used to identify a machine on a network. Each analyzer is assigned a unique Ethernet address at the factory and it is stored in the analyzer's ROM.

External trigger signal A TTL signal that is input to an analyzer and initiates a measurement sweep or similar event, making the measurements synchronous with the external triggering source.

F

Filter A passive device that allows some frequencies to pass and attenuates others, depending on the type and specifications. A high-pass filter passes frequencies above the cutoff frequency, a low-pass filter passes frequencies below the cutoff frequency, and a band-pass filter passes frequencies between two specific frequencies.

Firmware An assembly made up of hardware and instruction code. The hardware and instruction code is integrated and forms a functional set that cannot be altered during normal operation. The instruction

code, permanently installed in the circuitry of the instrument, is classified as ROM (read only memory). The firmware determines the operating characteristics of the instrument or equipment.

Flatness The amplitude and phase response of a device under test (DUT), a signal source, a receiver, or a combination of these. See also [Frequency Response](#).

FM Frequency Modulation

Frequency The number of periodic oscillations, vibrations, or waves per unit of time, usually expressed in cycles per second, or Hertz (Hz).

Frequency Accuracy The uncertainty with which the frequency of a signal or spectral component is indicated, either in an absolute sense or relative to another signal or spectral component. Absolute and relative frequency accuracies are specified independently.

Frequency Range The range of frequencies over which a device or instrument performance is specified.

Frequency Resolution The ability of a network analyzer to measure device characteristics at closely spaced frequencies and display them separately. Resolution of equal amplitude responses is determined by IF bandwidth. Resolution of unequal amplitude responses is determined by IF bandwidth and bandwidth selectivity.

Frequency Response The peak-to-peak variation in the displayed amplitude response over a specified center frequency range. Frequency response is typically specified in terms of dB, relative to the value midway between the extremes.

Frequency Span The magnitude of the displayed frequency component. Span is represented by the horizontal axis of the display. Generally, frequency span is given as the total span across the full display. Some analyzers represent frequency span (scan width) as a per-division value.

Frequency Stability The ability of a frequency component to remain unchanged in frequency or amplitude over short and long-term periods of time. Stability refers to an oscillator's ability to remain fixed at a particular frequency over time.

Front Panel Key Keys that are located on the front panel of an instrument. The key labels identify the function the key activities. Numeric keys and step keys are two examples of front panel keys.

Full 2-Port Calibration See [Error Correction, 12-Term](#).

Function The action or purpose that a specific item is intended to perform or serve. The network analyzer contains functions that can be executed via front panel key selections, or through programming commands. The characteristics of these functions are determined by the firmware in the instrument. In some cases, a DLP (downloadable program) execution of a function allows you to execute the function from front panel key selections.

Fundamental Frequency In any waveform, the lowest frequency component; all other components are harmonics. A pure sinusoid has only one component, the fundamental.

G

Gb Gigabit

GB Gigabyte

GHz Gigahertz

GIF Graphics Interchange Format - Standard graphic format to store bitmapped graphics files.

Giga Prefix for one billion.

GP I/O General Purpose Input / Output; a connector usually on the back of an instrument that allows communication with other test equipment, external test sets, switches, and computers that enable the instrument to be triggered or to trigger external equipment. An example is a foot switch that continues or cycles a measurement, allowing the operator to use both hands on the test hardware.

GPIB General Purpose Interface Bus - IEEE 488 bus is interconnect bus and protocol, allows linking of instruments and computer.

Graticule (or Grid) Enclosed area where waveform is displayed on instrument. Tick marks, on frame or axis, are a scaling aid for making visual measurements.

Group Delay A measure of the transit time of a signal through a DUT versus frequency. Group delay can be calculated by differentiating the DUT's insertion-phase response with respect to frequency. See also [AM Group Delay](#) and [Deviation from Linear Phase](#).

GUI Graphical User Interface

H

Hardcopy Paper copy of data.

Hardkey A front-panel key, which engages a single analyzer function or presents a single menu of softkeys.

Horizontal Reference See [Reference Level](#).

Horizontal Resolution The analyzer's ability to take closely spaced horizontal data points over the full sweep.

Host Computer A computer or device on a network that provides end users with services such as computation and database access and that usually performs network control functions.

Host Name A unique name that is used to identify each host machine on a network. The host name is directly linked to, and can usually be used in place of, the IP address. The user or the system administrator usually creates the host name.

HP Hewlett-Packard Company

HPGL Hewlett-Packard Graphics Language

HP-IB Hewlett-Packard Interface Bus. A parallel interface that allows "daisy chaining" of more than one device to a port on a computer or instrument. Interface protocol is defined in IEEE 488.2; equivalent to the industry standard GPIB.

HTTP HyperText Transfer Protocol: Used to carry World Wide Web (WWW) traffic.

Hue The dimension of color referred to a scale of perceptions ranging from red through yellow, green, and blue, and back to red. A particular gradation of color, tint, shade.

I

I/O Input/Output

I/O Path Input/Output Path

IEEE Institute of Electrical and Electronic Engineers

IF Intermediate Frequency: the frequency at which a signal is processed after mixing.

Impedance The ratio of voltage to current at a port of a circuit, expressed in ohms.

Initialize The process that assigns information locations to a disk to prepare the magnetic media to accept files.

Input A path intended for putting a signal into an instrument.

Most network analyzers have either 3 (labeled A, B, and R) or 4 inputs (labeled A, B, R1, and R2). Inputs are not the same as channels.

Input Attenuator An attenuator between the input connector and the first mixer of a spectrum analyzer (also called an RF attenuator). The input attenuator is used to adjust the signal level incident to the first mixer, and to prevent gain compression due to high-level or broadband signals. It is also used to set the dynamic range by controlling the degree of internally-generated distortion. For some analyzers, changing the input attenuator settings changes the vertical position of the signal on the

display, which then changes the reference level accordingly. In Agilent microprocessor-controlled analyzers, the IF gain is changed to compensate for changes in input attenuator settings. Because of this, the signals remain stationary on the display, and the reference level is not changed.

Insertion Loss The difference between the power measured before and after the insertion of a device. The attenuation between the input and output of a device.

Intensity Brightness; emitting or reflecting light; luminosity.

Interface A connection that allows a common communication link between two or more instruments.

Intermodulation Distortion Undesired frequency components resulting from the interaction of two or more spectral components passing through a device having nonlinear behavior, such as a mixer or an amplifier. The undesired components are related to the fundamental components by sums and differences of the fundamentals and various harmonics. The algorithm is: $f_1 \pm f_2$, $2xf_1 \pm f_2$, $2xf_2 \pm f_1$, $3xf_1 \pm 2x f_2$, and so on.

Internet The connection of two or more distinct networks. Often a gateway or router is used to make the connection.

Interpolate To determine a value of a signal between two adjacent points by a procedure or algorithm.

IP Internet Protocol

IP Address Internet protocol address: a unique number that is assigned to each device which is to be connected to a TCP/IP network. Before using an analyzer on a network, your network administrator will need to assign an IP address. An IP address consists of a 32-bit value presented in decimal dot notation: 4 octets (bytes) separated by a dot.

ISDN Integrated Services Digital Network: A standard digital service capability that features one or more circuit-switched communication channels capable of carrying digital voice, data, or image signals, a packet-switched channel for out-of-band signaling and control. In addition, ISDN provides a collection of standard and optional features that support information productivity for the user, providing higher-speed Internet access than analog systems.

ISO International Standards Organization

Isolation A specification or measure of the immunity that one signal has to being affected by another adjacent signal. The occurrence is known as crosstalk.

Isolator An RF device used for providing isolation between paths and components. Made from a 3-port circulator, the third port being terminated in a 50ohm load.

J

K

Kilo Prefix for one thousand.

KB Kilobyte

Kb/s Kilobytes per second

L

LAN Local Area Network

LANS Local Area Network System

LCD Liquid Crystal Display

LED Light Emitting Diode

LIF Logical Interchange Format (used for older HP disk drives/computers)

Limit Lines Lines input by the user that overlay the analyzer's measurement data to allow automatic detection of data that is out of the acceptable range. Pass/Fail annotation, audio alarms, or electronic output can be triggered to notify the operator or on-line computer program of the over-limit condition.

Limit-Line File The user-memory file that contains the limit-line table entries.

Limit-Line Table The line segments of a limit line are stored in the limit-line table. The table can be recalled to edit the line segments, then restored in the limit-line file.

Linear Device A device in which the output is continuously proportional to the input.

LO Local Oscillator. In a superheterodyne system, the LO is mixed with the received signal to produce a sum or difference equal to the intermediate frequency (IF) of the receiver.

LO Feedthrough The response that in a superheterodyne system when the first local oscillator frequency is equal to the first IF.

Load A one port microwave device used to terminate a path in its characteristic impedance.

Load Match A measure of how close the device's terminating load impedance is to the ideal transmission line impedance. Match is usually measured as return loss or standing wave ratio (SWR) of the load.

Local Lock Out A condition or command that prevents analyzer front-panel entries (and disables the

Local key).

Local Operation To operate manually from the front panel.

Log Logarithm

Log Display The display mode in which vertical deflection is a logarithmic function of the input signal amplitude. Log display is also called logarithmic display. The display calibration is set by selecting the value of the reference level position and scale factor in dB per division.

LRM Line-Reflect-Match. See [Calibration, TRL, and LRM](#).

M

Magnitude The amplitude of a signal measured in its characteristic impedance without regard to phase. See also [Scalar](#).

Marker A graphical symbol along a display trace that is annotated with measurement characteristics of that specific data point.

Marker Functions Mathematical or statistical computation on the data of one or more markers to provide the operator more information. For example, the marker delta function calculates and displays the difference between two markers.

Maximum Input Level The maximum signal power that may be safely applied to the input of an analyzer. The maximum input level is typically 1 W (+30 dBm) for Agilent spectrum analyzers.

MB Megabyte

Measurement Uncertainty The quantified amount of error in a measurement situation. Calibrations are intended to reduce the amount of uncertainty. The following are sources of measurement errors that lead to uncertainty:

- Systematic errors (imperfections in calibration standards, connectors, cables, and instrumentation)
- Random errors (noise, connector repeatability)
- Drift (source and instrumentation)

Mega Prefix for one million.

Memory A storage medium, device, or recording medium into which data can be stored and held until some later time, and from which the entire original data may be retrieved.

Memory Card A small memory device shaped like a credit card that can store data or programs.

Menu The analyzer functions that appear on the display and are selected by pressing front panel keys. These selections may invoke a series of other related functions that establish groups called menus.

MHz Megahertz

milli Prefix for one-thousandth.

Modem Modulator/Demodulator

Modulation The process, or the result of the process, of varying a characteristic of a carrier signal with an information-bearing signal, causing the carrier to contain the information. See [AM](#) and [FM](#).

Monitor Any external display.

Monochrome Having only one color (chromaticity).

ms Millisecond

mW Milliwatt: one thousandth of a watt

Multisync A type of monitor that can synchronize its horizontal sweep to various frequencies within a specified range.

N

Narrowband In network analysis, the frequency resolution of the analyzer's receiver that is sufficiently narrow to resolve the magnitude and phase characteristics of narrowband devices. The reduced receiver bandwidth usually decreases the noise floor of the receiver, providing more measurement amplitude range.

Narrowband Device A device whose transfer characteristics are intended to operate over a very narrow frequency range and are designed to provide well-defined amplitude responses in that range, such as a band pass filter.

Network Analysis The characterization of a device, circuit, or system derived by comparing a signal input going into the device to a signal or signals coming out from the device.

NIST National Institute of Standards and Technology

Nit The unit of luminance (photometric brightness) equal to one candela per square meter.

Noise Random variations of unwanted or disturbing energy in a communications system from man-made and natural sources that affects or distorts the information carried by the signal. See also [Signal-to-Noise Ratio](#).

Noise Figure (F): For a two-port device, a measure of how the noise generated inside the device degrades the signal-to-noise ratio of a signal passing through the device at 290 degrees, usually expressed in dB.

Noise Floor The analyzer's internal displayed noise. The noise level often limits how small a signal magnitude can be measured. In network analysis, noise floor is measured with the test ports terminated in loads, full two-port error correction, 10 Hz IF bandwidth, maximum test port power, and no averaging during the test.

Non-Insertable Devices In measurement calibration, a device that cannot be substituted for a [Zero-Length Through Path](#). It has the same type and sex connectors on each port, or a different type of connector on each port.

Nonvolatile Memory Memory data that is retained in the absence of an ac power source. This memory is typically retained with a battery. Refer also to battery-backed RAM.

Normalize To subtract one trace from another to eliminate calibration data errors or to obtain relative information.

O

Offset To move or set off a determined amount. Used in instruments for offsetting frequencies, limits, delay, loss, impedance, etc.

Output Attenuation The ability to attenuate the signal, the source, in order to control its power level.

P

PC Personal Computer

PDF Portable Document Format (used on the Web)

Parser, Command Reads program messages from the input queue of a device in the order they were received from the controller. The parser determines what actions the analyzer should take. One of the most important functions of the command parser is to determine the position of a program message in the analyzer SCPI command tree. When the command parser is reset, the next element it receives is expected to arise from the base of the analyzer command tree.

Peak Search A function on an analyzer that searches for the largest response and places a marker on it.

Phase The fractional part of a cycle through which an oscillation has advanced, measured from an arbitrary starting point; usually measured in radians or degrees. In network analysis, the phase response of the device under test is the change in phase as a function of frequency between the input stimulus

and the measured response.

Port The physical input or output connection of an instrument or device.

Port Extension Redefining the reference plane to other than that established at calibration. A new reference plane is defined in seconds of delay from the test set port.

Positive Peak The maximum, instantaneous value of an incoming signal.

Postscript (.ps files) Stores bitmapped graphics files in an encapsulated format for direct use by postscript printers.

Power, Max Input The upper limit to input power for which the specifications apply. Some specifications may have different levels of maximum inputs. For example, compression power maximum is usually higher than the harmonic distortion maximum.

Power, Safe Input The input power, usually in dBm, allowed without damaging the instrument.

Preset A pre-defined instrument state (that also runs an analyzer self-test). The action of pushing the Preset key.

Protocol A set of conventions that specify how information will be formatted and transmitted on a network, and how machines on a network will communicate.

Q

Q or Q Factor The ratio of energy stored to energy lost in a resonant circuit. High Q indicates a sharp resonance response over frequency.

Query Any analyzer programming command having the distinct function of returning a response. These commands may end with a question mark (?). Queried commands return information to the computer.

R

r + jx Expression for complex impedance, where r represents the resistive portion and x represents the reactive portion.

R Channel Reference Channel

RAM Random Access Memory, or read-write memory: A storage area allowing access to any of its storage locations. Data can be written to or retrieved from RAM, but data storage is only temporary. When the power is removed, the information disappears. User-generated information appearing on a display is RAM data.

ROM Read Only Memory: A storage area that can be read only; it cannot be written to or altered by the user. In instruments, the storage area that contains the "brains" or operational programming; the firmware.

Receiver A circuit or system designed for the reception and/or measurement of signals in a specified frequency spectrum.

Receiver Dynamic Range See [Dynamic Range](#).

Reference Level An instrument function that allows the user to set the amplitude value at the reference position. On network analyzers, the reference position is also selectable. On some spectrum analyzers, the reference position is fixed at the top of the display.

Reference Plane The electrical location at which a network analyzer assumes the system connectors and fixturing ends and the DUT begins. The reference plane is set by using calibration standards with known electrical length. The closer the reference plane is to the device under test (DUT), the better the characterization of the device because of the elimination of test system uncertainties.

Reference Receiver In a network analyzer, the receiver that measures signals as they come out of the source, before they are incident on the test port and DUT. Typically, these signals are used to compare with the signal at the Test Port Receiver, to determine the affect that the DUT has on the signal. In a 2-port network analyzer, these are typically named 'R1' (port 1) and 'R2' (port 2). [See a block diagram](#) of the receivers in your PNA.

Reflection The phenomenon in which a traveling wave strikes a discontinuity and returns to the original medium.

Reflection Coefficient The ratio of the reflected voltage to the incident voltage into a transmission line or circuit. If a transmission line is terminated in its characteristic impedance, the reflection coefficient is zero. If the line is shorted or open the coefficient is 1. See also [Return Loss](#) and [SWR](#).

Reflection Measurements Measurements that characterize the input and /or output behavior of the device under test (DUT). Measured as the ratio of the reflected signal to the incident signal as a function of frequency. Parameters are called return loss, reflection coefficient, impedance, and standing wave ratio (SWR), all as a function of frequency. See also [S-Parameters](#).

Remote A mode of operation where another device (or computer) controls an instrument via the HP-IB. In this mode, the instrument front panel keys are disabled. Front panel operation is called local operation.

Remote Programming The automatic operation of an instrument by a computer, usually through a HP-IB, LAN, or RS-232 link.

Resolution The ability of a receiver to resolve two signals.

Resolution Bandwidth The ability of a spectrum analyzer to display adjacent responses discretely (Hertz, Hertz decibel down). This term is used to identify the width of the resolution bandwidth filter of a spectrum analyzer at some level below the minimum insertion loss point (maximum deflection' point on the display). Typically, it is the 3 dB resolution bandwidth that is specified, but in some cases the 6 dB resolution bandwidth is specified.

Return Loss The amount of dB that the reflected signal is below the incident signal. If zero signal is reflected, the impedance of the device is equal to the characteristic impedance of the transmission system, and return loss is infinite. If the entire incident signal is reflected, the return loss is zero. See also [S-Parameters](#), [Reflection Coefficient](#), and [SWR](#).

Reverse Measurement The measurement of a device from output to input.

RF Radio Frequency (from approximately 50 kHz to approximately 3 GHz). Usually referred to whenever a signal is radiated through the air.

ROM Read Only Memory

S

S/N Signal-to-Noise Ratio

Sampler An electronic component that captures the signal level and phase across a known impedance at a uniform rate. In Network Analyzers, this sampling rate must be sufficiently high and precisely timed to make accurate measurements. Network analyzers typically have three or four samplers or mixers.

Sampler Bounce The leakage or crosstalk between a network analyzer's samplers. Delay in this crosstalk caused by leakage transmission propagation, give the interference its "bounce" appearance. Sampler bounce causes an increase in the noise level of the affected channel, reducing the sensitivity of the analyzer.

Saturation The degree of color purity, on a scale from white to pure color.

Scalar A quantity that has magnitude but no phase. A network analyzer capable of measuring only magnitude.

Scale Factor The display vertical axis calibration in terms of units per division.

SCPI Standard Commands for Programmable Instruments

Screen The physical surface of the CRT or flat panel upon which the measurement results, setup information, softkey definitions, and other instrument communication is presented.

Self-Test A group of tests performed at power-up (or at preset) that verify proper instrument operation.

Sensitivity The minimum input signal required to produce a specified output signal having a specified signal-to-noise ratio, or other specified criteria.

On a spectrum analyzer, the level of the smallest sinusoid that can be observed, usually under optimized conditions of minimum resolution bandwidth, 0 dB input attenuation, and minimum video bandwidth.

The normalized change in YIG component's center frequency resulting from a change in tuning coil current, specified in MHz/mA.

Serial Prefix The five-character prefix that begins an instrument serial number; used to represent versions of firmware or hardware changes that have occurred.

Server A device that is configured to provide a service to other devices on a network, such as shared access to a file system or printer.

Signal-to-Noise Ratio SNR: The ratio of the amplitude of the desired signal to the amplitude of noise signals, usually expressed in dB and in terms of peak values for impulse noise and root-mean-square values for random noise.

Single Sweep Mode The spectrum analyzer sweeps once when trigger conditions are met. Each sweep is initiated by pressing an appropriate front panel key, or by sending a programming command.

Small Signal Gain Compression A situation when the input signal's measured amplitude is less than its actual level due to overloading of the network analyzer's input mixer; the analyzer is operating nonlinearly. For broadband analyzer detectors, a signal other than the one under test can put the analyzer into this gain compressed mode, thereby making even lower level signals appear at a lower level than actual. The broadband mode measures all the power incident to the analyzer, not just the signals at the frequency of interest.

Smith Chart A graphical mapping of the complex reflection coefficient into normalized complex impedance. Circles on the chart represent constant resistance and radiating lines orthogonal to the circles represent constant reactance. The center of the chart represents the characteristic impedance of the transmission system. Any point on the chart defines a single complex impedance. A line on the chart represents changing impedance over frequency.

SOLT Short-Open-Load-Through calibration. See also [Calibration, SOLT](#).

Source A device that supplies signal power; a sweep oscillator or synthesized sweeper.

Source Amplitude Accuracy The amplitude uncertainty, in dB, of the source power readout.

Source Amplitude Flatness The amplitude flatness, in dB, of the source power over the frequency

range specified.

Source Frequency Resolution The smallest unit of frequency which can be set and/or measured, in Hz.

Source Frequency Time Base Accuracy A measure of the analyzer's frequency stability measured in parts per million (ppm. or 1 part in 10E6). For example, a stability of ± 5.0 ppm means that an analyzer will measure 1 MHz to an accuracy of $\pm 5 \times 10^{-6} \times 10^6 \text{ Hz} = \pm 5 \text{ Hz}$.

Source Frequency Time Base Stability A measure of the analyzer's time base accuracy over time and temperature. Typically the time base accuracy will be specified for 1 year. A typical temperature frequency stability is ± 10 ppm for $25^\circ \text{C} \pm 50^\circ \text{C}$.

Source Harmonics The level of harmonics generated by the analyzer's signal source, in dBc from the fundamental.

Source Match A measure of how close the signal source impedance is to the ideal transmission line impedance of the test system. Match is usually measured as return loss or standing wave ratio (SWR) of the source.

Span The stop frequency minus the start frequency. The span setting determines the horizontal-axis scale of the analyzer display.

Span Accuracy The uncertainty of the indicated frequency separation of any two signals on the display.

S-Parameters (Scattering Parameters) A convention used to characterize the way a device modifies signal flow using a network analyzer. A two port device has four S-parameters: forward transmission (S21), reverse transmission (S12), forward reflection (S11), and reverse reflection (S22).

Stop/Start Frequency Terms used in association with the stop and start points of the frequency measurement range. Together they determine the span of the measurement range.

Storage States The number of settings, programs, traces, and other parameters available to be saved, cataloged, and recalled at any one time.

Storage, Disk An internal or external digital storage disk for saving test data, instrument settings, IBASIC programs, and other measurement parameters. Storage formats include MS-DOS (R) and HPs standard LIF with binary, PCX, HP-GL, or ASCII data formats.

Structural Return Loss Poor return loss in cable due to a periodic fault such as a periodic dent caused by dropping the cable spool or by the cable pulling process during manufacture.

Supplemental Characteristics Typical but non-warranted performance parameters, denoted as "typical", "nominal" or "approximate".

Sweep The ability of the source to provide a specified signal level over a specified frequency range in a specified time period. Also see [Sweep Mode](#) and [Sweep Type](#).

In data processing mode, a series of consecutive data point measurements, taken over a sequence of stimulus values.

Sweep Mode The way in which a sweep is initiated or selected, e.g., single, continuous, alternate, or chopped.

Sweep Type The method of sweeping the source, e.g., linear, log, or frequency step.

Sweeper A signal source that outputs a signal that varies continuously in frequency.

SWR Standing Wave Ratio, calculated as $(1 + \pi) / (1 - \pi)$ where π is the reflection coefficient.

Sync Synchronization, or Synchronized

Syntax The grammar rules that specify how commands must be structured for an operating system, programming language, or applications.

System Dynamic Range The difference between the maximum receiver input level and the receiver's noise floor. System dynamic range applies to transmission measurements only, since reflection measurements are limited by directivity.

T

T/R See [Transmission/Reflection](#).

Termination A load connected to a transmission line or other device.

Test Limit The acceptable result levels for any given measurement.

Test Port See [Port](#).

Test Port Receiver In a network analyzer, the receiver directly behind the test ports, used to measure the signal as it is reflected off, or transmitted through, the DUT. This signal is typically compared with the signal at the [Reference Receiver](#) to determine how the DUT affects a signal. In a 2-port network analyzer, these are typically named 'A' (port 1) and 'B' (port 2). [See a block diagram](#) of the receivers in your PNA.

Test Set The arrangement of hardware (switches, couplers, connectors and cables) that connect a test device input and output to the network analyzer's source and receiver to make s-parameter measurements.

Third Order Intercept TOI: The power input to a non-linear device that would cause third order

distortion at the same power level. TOI is a measurement to determine the distortion characteristics of a mixer or receiver. The higher the value, the more immune the receiver to internal distortion.

Thru Through line: A calibration standard. See [Calibration, SOLT](#).

Tint A shade of color; hue.

Toggle To switch states, usually to change a function from on to off, or off to on.

TOM Thru-Open-Match: A Rohde&Schwarz term to describe a calibration method.

Trace A series of data points containing frequency and response information. The series of data points is often called an array. The number of traces is specific to the instrument.

Tracking The ability of the analyzer's receiver to tune to the source frequency over the measurement frequency range. Poor tracking results in amplitude and phase errors due to the receiver IF circuits attenuating and delaying the device under test output.

Transfer Function The ratio of the output signal to the stimulus signal, both as a function of frequency.

Transmission See [Transmission Measurements](#).

Transmission Intermodulation Spurious A measure of the capability of the transmitter to inhibit the generation of intermodulation distortion products. Intermodulation spurious is sometimes called intermodulation attenuation.

Transmission Measurements The characterization of the transfer function of a device, that is, the ratio of the output signal to the incident signal. Most common measurements include gain, insertion loss, transmission coefficient, insertion phase, and group delay, all measured over frequency. See also [S-Parameters](#).

Transmission/Reflection (T/R) Refers to the suite of measurements made by a scalar or vector network analyzer to characterize a device's behavior over frequency. See also [S-Parameters](#).

Transparent Something that is not visible to the user. Usually a procedure that occurs without the user's initiation or knowledge.

Trigger A signal that causes the instrument to make a measurement. The user can select several options for triggering, such as manual, continuous, or external (for synchronizing measurements to an external source).

TRL Through-Reflect-Line. See [Calibration, TRL and LRM](#).

TTL Transistor-Transistor Logic

Two-Port Error Correction See [Error Correction, 12-Term.](#)

U

Uncorrected Measurements made without performing error correction.

Uncoupled Channels Stimulus or receiver settings allowed to be set independently for each channel.

UNI User-Network Interface: The point at which users connect to the network.

Units Dimensions on the measured quantities. Units usually refer to amplitude quantities because they can be changed. In analyzers with microprocessors, available units are dBm (dB relative to 1 mW dissipated in the nominal input impedance), dBmV (dB relative to 1 mV), dBW (dB relative to 1 W), V (volts), W (watts).

V

Variable A symbol, the value of which changes either from one iteration of a program to the next, or within each iteration of a program.

Vector A quantity that has both magnitude and phase.

A network analyzer capable of measuring both magnitude and phase.

VEE Visual Engineering Environment (Agilent software product)

Velocity Factor A numerical value related the speed of energy through transmission lines with different dielectrics (.66 for polyethylene). Used in making time domain measurements.

Vertical Resolution The degree to which an instrument can differentiate amplitude between two signals.

Video An electrical signal containing timing, intensity, and often color information that, when displayed, gives a visual image.

Video Bandwidth In spectrum analyzers, the cutoff frequency (3 dB point) of an adjustable low-pass filter in the video circuit. When the video bandwidth is equal to or less than the resolution bandwidth, the video circuit cannot fully respond to the more rapid fluctuations of the output of the envelope detector. The result is a smoothing of the trace, or a reduction in the peak-to-peak excursion, of broadband signals such as noise and pulsed RF when viewed in broadband mode. The degree of averaging or smoothing is a function of the ratio of the video bandwidth to the resolution bandwidth.

Video Filter In spectrum analyzers, a post-detection, low-pass filter that determines the bandwidth of the video amplifier. It is used to average or smooth a trace. Refer also to [Video Bandwidth](#).

W

Waveform A representation of a signal plotting amplitude versus time.

Wireless A term that refers to a broad range of technologies that provide mobile communications for home or office, and "in-building wireless" for extended mobility around the work area, campus, or business complex. It is also used to mean "cellular" for in-or out-of-building mobility services.

WWW World Wide Web

X

Y

Z

Zero-Length Through Path In a measurement calibration, when the two test cables mate together directly without using adapters or a thru-line. See also [Non-Insertable Devices](#).

INDEX

. .csa - .cst - .sta - .cal files	607
.cti files	607
.mdf files	607
.mxr file	
.NET	2554
.s1p - .s2p - .s3p files	607
.xml files	3582
1	
10 MHz adjustment	4125
10 MHz reference	111
1-port calibration; reflection	327
2	
2-port calibration; isolation portion	511
2-port devices reflection accuracy	299
3	
3-Port error terms	497
3-port SOLT calibration	327
4	
4-port Cal using 2-port ECal	403
4-port embed-deembed	453
4-port error terms	497
4-port on wafer calibrations	
4-port with external testset	4225
8	
81110A Pulse Generator	3892
82357A	4083
8510 cal kits - importing	532
8510 data processing mode	581
A	
about Network Analyzer	162

absolute output	4267
absolute output power	4267
accessories- list of	4076
accessories, effects of	314
accuracy of calibration	511
accuracy, improving	
accuracy, on low-loss 2-port devices	299
accuracy, phase measurements	289
accuracy, reflection	299
accurate measurement calibrations	511
active channel	119
active data trace	581
active probes	4270
adapter removal cal - ECal	515
adapter removal cal - mechanical stds	515
adapters - using to calibrate	515
adaptor - characterize macro	478
ADC Receiver Measurements	278
add users	87
address - GPIB	3530
adjust the display	260
adjustments	
administrative tasks	
administrator - user	87
administrator password	87
admittance chart	231
AGC loops, testing amps with	4317
Agilent IO libraries	3573
AgileUpdate	4085
ALC leveling	188
aliasing	3965
aliasing	3965
all off	260

alternate sweep, crosstalk	313
amplifier measurement- high-gain	4315
amplifier parameters	4277
AM-to-PM conversion	4272
analog receivers	278
analyzer specifications	4331
annotation - display	260
antenna features	3649
antivirus protection	99
aperture, group delay	4308
app notes	4250
application crashes	3985
applying cal sets	363
arbitrary impedance	547
arbitrary ratio	170
arbitrary segment sweep	206
arbitrary Z standard	527
archiving cal sets	363
arranging windows	244
ASCII file types	607
assignments - cal class	556
attenuation	188
attenuation-source	188
attenuators, receiver	188
auto port ext.	469
auto save	607
autocheck	4085
autostart PNA	116
auxiliary front-panel jumpers	
auxiliary trigger	223
averaging, sweep	304
how to set	304
avoid spurs	3654

B		
	balanced measurements	4283
	banded millimeter modules	4162
	banded test heads	4162
	bandwidth	563
	beeping - stop	90
	bias input	111
	binary file types	607
	blanking - frequency band crossings	179
	block diagram - configurable test set	4331
	block diagram - standard test set	4331
	booster amp with SMC	3709
	boot from the recovery partition	96
	broadband LO tuning	3738
C		
	C and COM	2550
	cables - front-panel jumpers	
	cables - how to ground	4266
	cal all channels	353
	cal kit manager	532
	cal set viewer	497
	cal sets	363
	calibrate touchscreen	
	calibration kits	527
	creating	541
	databased	511
	modifying	532
	waveguide	544
	calibration select	327
	calibration standards - measuring	521
	accurate	511
	cal set	363
	Cal Wizard	331

class assignments	556
converter	3654
data-based model	511
electronic (ECal)	377
enhanced response	327
instrument calibration and verification	4105
isolation (crosstalk)	313
mixer	3654
model	511
non-insertable device	515
overview	494
polynomial model	511
power calibration	433
preferences	140
select a,	327
standards	527
types	327
using thru adapters	515
validity of,	521
window	331
CalPod	410
care of connectors	4255
channel settings	119
channels - copy	271
channels - traces and windows	119
characterize adapter macro	478
characterizing the cal mixer	3712
citifiles	607
class assignment	527
class assignments - modify	556
classes calibration - about	527
classes-measurement	167
clean connectors	4255

code translator application	
coefficient, reflection	4326
coefficient, transmission	4333
collections	2536
COM Data Types	2538
COM Events	2543
COM Examples	
COM Fundamentals	2528
COM Object Model	636
COM versus SCPI	3643
combiner	248
COM-DCOM configuration	2523
common mode	4283
compensating for test fixture	453
complex impedance	4277
compression, gain	3775
computer name	90
conductive table mat	4266
confidence check	521
configurable test set	
block diagram	4331
dynamic range	284
options	4091
configuration - millimeter module	4162
configuration - PNA standard	4091
configuration IF path	4152
configuration RF path	248
Configure an External Device	4180
configure an external LO source	4180
configure for COM-DCOM	2523
configure for GPIB-SICL	3540
connect to disc drive	628
connectivity	139

connector care	4255
connector repeatability	497
connectors - test port	4091
continuous	218
conversion-loss measurement	3760
Copy Channels	271
corrected system performance - specifications	4331
correction level	372
correction ON/off	372
corrupted operating system	96
counter averaging	304
coupled markers	563
coupled trace settings - time domain	3965
coupling between ports	188
CPU speed	90
crash - PNA	3985
creating a noise fig meas	3838
creating windows	119
crosstalk specification	4331
crosstalk, receiver	313
cti files	607
customize your analyzer screen	260
status bar	260
tables	260
title bars	260
toolbars	260
CW time sweep	206
cXL code emulation	
D	
data display	231
data points	287
data processing - 8510 mode	581
data processing chain	

data saving	607
data trace display	581
data transfer speed	321
data transfer time	4331
data transfer time - specifications	4331
data-based model for calibration	511
date and time	90
DC control	4192
DC source in PNA	278
default conditions	165
define data saves	607
definitions	4331
definitions - specs	4331
defragmenter - disk	94
delay functions	4295
delay, electrical	289
delay, group	4308
accuracy considerations	4308
aperture, what is	4308
what is	4308
why measure	4308
delta marker	563
delta match cal	489
detector inputs - specifications	4331
deviation from linear phase	4277
deviation from linear phase	4297
accuracy considerations	4297
using electrical delay	4297
what is	4297
why measure	4297
device connections	302
device delay	296
device-side USB	3562

diagnostic tools and adjustments	
Dialog transparency	152
dielectric - velocity factor	289
differential mode	4283
dimensions-physical	4331
directivity error	497
disaster recovery partition	96
discrete marker	563
disk defragmenter	94
display	260
display formats	231
display screen	
display settings	
display specifications	4331
display test	4127
distance marker settings	3965
documentation - PNA	154
documentation warranty	154
domain	563
drift	302
errors	497
frequency	302
temperature	302
drive mapping	628
DSP Number	162
during retrace and sweep	188
dwel time - segment sweep	206
dwel time - stepped sweep	206
dynamic accuracy specification	4331
dynamic range	284
E	
E5091 Testset	4221
ECal	377

4-port Cal using 2-port ECal	403
confidence check	521
models	4076
procedure	377
thru methods	515
user-characterization	387
edge triggering	223
editor - equation	587
effects of accessories	314
electrical delay	289
electrical delay, using	4297
electrical length, add	296
electrically-long device measurements	296
electronic calibration (ECal)	494
electrostatic discharge protection	4266
embed and De-Embed a fixture	453
embedded LO meas	3738
EMI specification	4331
emulation - code	
enhanced response cal	327
ENR file	3878
entry toolbar	
environment specifications	4331
equal length adapter method	487
equation editor	587
erase memory	134
error correction	372
no correction	372
potential degradation	372
validation calibration accuracy	521
error messages - about	4072
error messages - list of	3989
error terms - 4 port	497

error terms-monitoring	497
errors, measurement	497
errors, random	497
errors, systematic	497
ESD protection	4266
ethernet	111
EULA - Microsoft	66
event log	3985
examples of cal set usage	363
examples-programming	635
excursion	563
execute	563
expanded math	556
extensions, port	511
external CD-RW drive, save file to	607
external DC meter and source	4192
External Device	4180
external monitor - VGA	
external PC drive sharing	628
external Pulse Gen config	4211
external source configuration	4180
external source synchronization	4216
external testset control	4225
external testset io connector	3588
external trigger	223
F	
factor - velocity	289
fast groups	3649
fast segments	3649
fast sweep	206
FastCW	3649
favorites	
FCA	3654

FIFO buffer	3649
file recall	607
file save	607
file types	607
files - manage without a mouse	607
firmware and help upgrade	4085
firmware rev number	162
fixed marker type	563
fixture simulator	453
flatness and gain, small signal	4300
flush thru cal	515
FOM	3745
formats - data	231
group delay	231
imaginary	231
linear mag	231
log mag	231
phase	231
polar	231
real	231
rectangular	231
smith chart	231
SWR	231
forward and reverse SMC meas	3669
frequency band crossings	179
frequency blanking	134
frequency converter application	3654
frequency drift	302
frequency drift	302
frequency offset	3745
frequency range	179
PNA models	4091
frequency reference, 10 MHz	111

frequency resolution	179
frequency, CW	179
front panel	4331
front panel jumpers	
dynamic range	284
option014	
specifications	
front panel keypad not working	3985
front panel specification	4331
front panel tour	101
full computer name	90
functions, marker	563
G	
gage connectors	4255
gain and flatness, small signal	4300
gain compression app	3775
gain definition	4277
gain drift versus time	4277
gain drift versus time, definition	4277
gain flatness	4277
gain flatness, definition	4277
gating	314
gating	3965
reflection response	314
transmission response	314
general specifications	4331
getting data with scpi	3544
getting started	
global - trigger	218
global delta match cal	489
global pass-fail indicator	602
GPIB - configure	3540
GPIB Address	3530

GPIB Commands	2556
GPIB Examples	
GPIB Fundamentals	3530
GPIB Interface	4083
GPIB Local - Remote Label	3540
GPIB Pass-Through	3398
grid lines solid/dotted	260
grid, screen	
grounding test cables	4266
group delay format	231
group delay measurement	4308
group trigger	218
guided calibration	331
guided power cal	348
H	
H08 option	3892
H08 specifications	4331
H85 option	4317
handle to an object	2532
handler io connector	3595
handshake - trigger	223
hard disk - version	162
hard disk problems - preventing	
hard disk recovery	96
hard disk, save file to	607
hard drive - removable	111
hard drive defragmenter	94
harmonics specification	4331
HDD capacity	90
HDD Problems	
heel strap - ESD	4266
help - about	162
help -using	154

high power measurements	4317
high-gain amplifier measurements	4315
hold - max & min	587
hold mode	218
host ID	4101
hot s22	
HPIB Commands	2556
I	
IF access (H11)	4152
IF bandwidth	304
IF bandwidth	304
IF bandwidth - for segment sweep	206
IF bandwidth - how to set	304
IF gain - phase lock- adjustment	4132
IF gain configuration	4152
IF Path Config (PNA-X)	4152
IF switch configuration	4152
IM Spectrum	3935
image - saving screen	629
imbalance parameters	4283
IMD	
impedance	170
setting system	4248
impulse and step modes	3965
Incident power - SMC	3669
independent segments	206
input	4331
input receiver - specifications	4331
input, 10 MHz reference	111
inspect connectors	4255
Installation and Quick Start Guide	154
installed options	4091
instrument calibration	4105

instrument noise	497
instrument specifications	4331
instrument state	607
Integrated Pulse App	3816
interface control	3582
intermod distortion	
interpolation ON/off	372
interpolation, effects on error correction	372
inverse Smith chart	231
IP address	90
isolation error	497
isolation, 2-port calibration	511
isolation, reverse	4323
J	
jpeg format	629
jumper cables - front-panel	
K	
keypad not working	3985
knob, front panel	101
L	
LAN Connections	99
LAN connector	111
LCD display test	4127
leakage, crosstalk	313
level	188
leveling - receiver	199
leveling-ALC	188
limit lines	602
limit table	260
linear frequency sweep	206
linear phase shift, compensate for	289
linear phase; deviation from	4297
LO - embedded in mixer	3738

LO source - configure for mixers	4180
load match error	497
load types	547
local	3540
log errors	4072
log on	87
log sweep	206
logical receiver notation	170
long devices electrically	296
loops - front-panel jumpers	
loss	170
lost phase lock	3985
low-loss 2-port devices	299
LRL line auto characterization	559
LVL	188
LXI	148
M	
macro - characterize adaptor	478
macro/local key	101
macros	322
magnitude offset	238
making noise settings	3838
manage files without a mouse	607
manual	218
map - data access	
mapping a drive	628
marker functions	563
marker search	563
bandwidth	563
domain	563
execute	563
peak	563
target	563

types	563
markers	563
coupled	563
delta	563
discrete	563
display	563
distance settings - time domain	3965
drag	
fixed	563
formats	563
move	563
normal	563
readout	563
reference	563
table	563
mask	3965
masking	3965
material handler interface control	3582
material handler io connector	3595
math	581
math operations, data-memory	581
statistics	581
mating plane surfaces	4255
max - min hold	587
measurement annotation	260
measurement calibration	494
measurement calibration method	327
accurate	511
improve crosstalk	313
select type	327
validity of	521
measurement classes	167
measurement errors	497

measurement parameters	170
arbitrary ratio	170
balanced	170
S-parameters	170
unratioed power	170
Measurement receiver	4176
measurement reference plane	511
measurement sequence	133
measurement setups, preconfigured	244
measurement stability, increase	302
measurement throughput	4331
measurement throughput - specifications	4331
measurement throughput techniques	317
measurement, averages	304
measurements, phase	4319
mechanical stds calibration	331
memory	581
memory - random access	90
memory trace	581
messages - error	4072
Millimeter Module Configuration	4162
minimize application	260
mixed mode S-parameters	4283
mixer testing	
mixer with embedded LO	3738
model	230
model for data-based calibration	511
model for polynomial calibration	511
data-based	511
polynomial	511
modify	532
modify calibration kits	532
monitors, external	

mouse - manage files without	607
move marker	563
multiple measurements for throughput	317
multiple power sensors	348
multiple standards	331
multiport testset - E5091A	4221
multiport testset control	4225
N	
N44xx testsets	4225
n5230 models	3595
N5260A	4162
N5264A	4176
name of computer	90
NET	2554
Network Analyzer - about	162
network analyzer specifications	4331
network analyzer users	87
network drive mapping	628
new features	57
new windows	119
noise - reduce	304
Noise Cal	3878
noise correlation matrix	3838
Noise Figure App	3838
noise floor	313
noise reduction	304
noise source	3878
noise specification	4331
nominal incident power	3669
non-insertable device calibrations	515
normal	563
normal marker type	563
normalize	581

number of points	287
O	
object hierarchy	2532
obsolete COM commands	
offset - frequency	3745
offset - magnitude	238
offset - phase	289
offset load	547
offset-receiver atten	188
offset-source atten	188
on and off	116
on-wafer 4-port cal	
open-loop leveling	188
operating system errors	4072
operating system recovery	96
operator's check	4128
optimum	188
option 008	3816
option 010, time domain	3965
option 014 or 015 - configurable test set	
option 082 SMC	4091
option 083	4091
option 088	252
option 108	4176
option 118	3649
option 550	4225
option 551	4225
option 897 & 898	4091
option enable	4101
option H08	4331
option H11	4331
option H29	3838
options available	4091

options installed	4091
output	4331
output power - SMC	3669
output power, absolute	4267
output, 10 MHz reference	4125
overlapping sweep segments	206
overlay windows	244
override - stimulus CW	3745
overview	3965
P	
parameters, amplifier	4277
parameters, measurement	170
pass-fail indicator	602
pass-through - GPIB	3398
password	87
path configuration IF	4152
path configuration RF	248
pc - connect to	139
peak	563
peak, marker	563
permission to launch and access PNA	2523
phase coherent measurements	294
phase control	252
phase lock lost	3985
phase measurement accuracy	289
phase measurements	4319
phase offset	289
phase offset	289
Phase Reference Cal	3692
phase shift	289
phase, linear	4297
phase-lock if gain adjustment	4132
phase-shift component	4308

PIA	2554
pixel test	4127
PMAR	4200
PNA autostart	116
PNA specifications	4331
PNA start	165
PNA-X	101
PNOP markers	563
point	218
point averaging	304
point spacing - x axis	206
point sweep	206
point-averaging	304
point-in-pulse	3892
points, number of	287
polar format	231
polynomial model for calibration	511
port extensions	469
port extensions	469
power calibration	433
power compensation - fixture	453
power I/O connector	3611
power level	188
power meter as receiver	4200
power On and Off	116
power sensor as receiver	4200
power unlevelled	188
power, line specifications	4331
power-up conditions	165
precise LO tuning	3738
preferences - cal	428
preferences - error	4072
preferences - PNA	140

preset conditions	165
preset user defined	165
Preventing HDD problems	
primary GPIB address	3530
Primary Interop Assembly	2554
print Help documentation	154
print options	629
print to file	629
printers	629
probe measurements	4270
probe power - PNA-X	101
probe specifications	4331
processing data map	
processor - CPU speed	90
product support	4119
programming guide	635
protection - antivirus	99
protection - ESD	4266
PSAT markers	563
PSG synchronize	4216
pulse I/O connector	3609
pulsed application	3816
Q	
queues	3544
Quick Start dialog	127
Quick Start Guide	154
quiet beeping	90
R	
RAM	90
random measurement errors	497
range, dynamic	284
range, frequency	4091
range, power	188

ratio, arbitrary	170
ratio, standing wave	4326
reading data	3544
rear panel	4331
rear panel interface control	3582
rear panel specifications	4331
rear panel tour	111
recall and sweep speeds - specifications	4331
recall measurements quickly	317
receiver attenuation	188
receiver attenuation	188
receiver calibration	4148
receiver crosstalk	313
receiver display	4151
receiver dynamic range	4331
receiver input - specifications	4331
receiver leveling	199
receiver power cal	433
receivers - number of reference	4091
receivers- measurement	170
recovering the operating system	96
redo	281
reduce trace noise	304
reference channel jumper	
reference impedance for a fixture	453
reference level	238
reference level & stability	4331
reference level & stability - specs	4331
reference marker	563
reference mixer	3712
reference plane	511
reference position	238
reference receivers	170

reference switch - port1	3745
reference, 10 MHz	111
reflection accuracy	299
reflection classes, calibration standards	532
reflection measurements	4326
reflection tracking error	497
relative velocity	289
remote-GPIB	3540
replacement COM commands	
resolution	3965
response calibration	327
restart - trigger	218
restore ECAL data	387
restore the operating system	96
retrace - power level during	188
return loss	4277
reverse isolation	4323
reverse isolation	4277
reverse sweep	206
revision number - PNA	162
rf probes	4270
right-click functions	
rolloff, coupler	314
S	
s1p - s2p -s3p files	607
save and recall a file	607
save as	607
save calibration	331
saving data	607
saving screen capture	629
scalar mixer calibration	3669
scale	238
scan disk	94

scope	218
scope, trigger	218
SCPI Commands	2556
SCPI Errors	4108
SCPI Examples	
SCPI Fundamentals	3530
SCPI Syntax	3535
screen arrange	244
screen capture	629
screen test	4127
screen, display	
search, marker	563
second source - Internal	137
securing the PNA	134
security frequency blanking	134
segment	260
segment point spacing	206
segment sweep	206
segment table	260
segments, limit	602
sequence, basic measurement	133
serial number - instrument	162
service request	3564
service routines	
setup macros	322
sharing a drive	628
Shut down PNA remotely	3648
SICL	3573
signal purity specification	4331
Simcal	3556
simulator - fixture	453
single	218
single and double click option.	99

single-ended devices	4283
sliding load	547
slope	188
slope, power	188
small signal gain and flatness	4300
SmartCal	331
SMC	3669
SMC phase	3686
smith chart	231
smoothing - trace	304
sockets	3540
softkeys	
software version	162
SOLT calibration	327
source	218
source - external synch	4216
source adjustment - service	4146
source attenuation	188
source match error	497
source phase control	252
source power cal	433
source unlevelled	188
source, trigger	218
spacing between frequency points	289
S-parameters	170
speaker	90
specifications	4331
speed, data transfer	321
speed, sweep	315
split windows	244
spreadsheet - saving data to	607
spurious specification	4331
spurs - avoid	3654

SRQ	3564
stability, measurement	302
stack windows	244
standard definition	527
standards, calibration	532
start PNA application	165
state - instrument	607
state, trigger	218
static electricity	4266
statistics	581
status bar	260
status registers	3564
stepped sweep	206
stimulus sweep	206
support	4119
swap equal adapter cal	487
sweep	206
sweep averages	304
sweep in reverse	206
sweep indicator	206
sweep setup	206
sweep speed	315
sweep time	206
sweep types	206
Swept IMD	3918
switch - port 1 reference	3745
switch repeatability	497
SWR	170
synchronize an external source	4216
synchronizing commands	3549
system bandwidths	4331
system bandwidths - specifications	4331
system impedance	4248

system performance, corrected	4331
system performance, uncorrected	4331
system restoration	96
system verification	4135
systematic measurement errors	497
T	
table mat - ESD	4266
tables	260
technical specifications	4331
technical support	4119
telnet	3540
temperature drift	302
temperature drift	302
terminate reflection measurement	299
test port input (receiver)	4331
test port output (source)	4331
test port specifications	4331
test set (configurable) block diagram	4331
test set block diagram	4331
test set interface control	3582
test set IO connector	3588
test set reference switch	3745
testset control - N44xx	4225
testset control - other	4225
testset control -E5091A	4221
theme - windows	99
threshold	563
throughput, increased	317
thru - unknown	515
thru response calibration	327
time and date	90
time domain	3965
timeout	3549

title - trace	260
title - window	260
title bars	260
TMSA	4283
toolbars	260
topology - balanced	4283
torque wrench	4255
touchscreen	
touchstone format	607
trace	119
trace coupling	3965
trace noise	4331
trace smoothing	304
trace status	260
trace triggering	218
traces - channels and windows	119
transfer speed, data	321
transferring data	3544
translator - code	
transmission	170
transmission classes, calibration standards	532
transmission tracking error	497
transparent dialogs	152
trigger	218
TRL calibration	405
troubleshoot the analyzer	3985
True Mode Stimulus App	3826
tuning for embedded LO	3738
type library	2523
types, marker	563
typical cycle time	4331

U

uncertainty curve equations	
-----------------------------	--

uncorrected system performance	4331
uncorrected system performance - specs	4331
under-sampled data	302
undo	281
unknown thru cal	515
unleveled	188
unratioed	170
unratioed power	170
unratioed power	170
unwrapped phase display	231
upgrade options	4091
upgrade PNA firmware	4085
uploading error terms	3556
USB device - the PNA	3562
USB Devices - remote control of	3571
USB hub	111
USB port specifications	4331
USB Power Sensor as Receiver	4200
USB Power Sensors	433
USB/GPIB Interface	4083
use nominal incident power	3669
User Key	
user logon	87
user name	87
user preset	165
user range	581
user span	563
user-characterization	387
V	
vector mixer calibration	3712
VEE RunTime - Agilent	3577
velocity factor	289
verification - system	4135

verifying calibration	521
version - firmware	162
VGA output connector	
Video Graphics Adapter	
view/modify standards	556
VISA	3573
VMC	3712
VMC with embedded LO	3738
VNC	146
W	
wake PNA at specified time	97
warm-up spec	4331
warranty period	4091
waveguide	544
web server	148
wide trace preference	140
Wincal	3556
window - calibration	331
window - customize	260
windowing	3965
windowing time domain	3965
windows - creating and managing	119
Windows 7 file locations	151
windows theme	99
windows traces and channels	119
Windows XP / 2000	99
windows, arranging feature	244
wizard, calibration	331
workgroup	2523
X	
X-axis display for FCA	3654
X-axis point spacing -segment sweep	206
XP - Windows	99

Y		
	Y-axis Scale	238
Z		
	Z conversion - balanced	453
	Z0 System Impedance	4248
	Z5326A H08 test set commands	3582
	zoom-trace	
