Intercomunicador Digital de Imagem e Som sem Fios

Autores: Sérgio Ivan Lopes Nº 18238
Fernando Jorge Ventura Nº 13246

Orientadores: Prof. Rui Escadas
Prof. Armando Pinto

Resumo

A aplicação que se pretende implementar, é um intercomunicador de som e imagem, portátil e de baixo custo. As comunicações são sustentadas por um sistema de comunicações local, LAN (Local Área Network), sem fios. Esta LAN permite que haja fluxo de informação entre os vários nós que a compõem (Câmaras, Displays, PC's). O custo do equipamento é um dos factores que se pretende controlar, por isso considerou-se o uso de um Gameboy™ como terminal de recepção de imagem, dado o seu baixo custo e a sua portabilidade. No final o equipamento desenvolvido terá uma relação qualidade/preço tentadora. Não é intenção deste projecto, implementar uma aplicação que apresente imagem de alta resolução e som de alta-fidelidade, o conceito base prende-se com uma comunicação fiável e eficaz, entre dois ou mais elementos do sistema.

Índice

Lista de Figuras

Lista de Tabelas

1. Introdução

No âmbito deste projecto pretende-se implementar um intercomunicador de som e imagem, autónomo, móvel e de baixo custo. Um intercomunicador, numa configuração básica, não é mais do que um conjunto de dois elementos que têm como função, trocar informação dum ponto para o outro. O objectivo principal é desenvolver, o hardware e respectivo software, para uma configuração básica, do tipo emissor-receptor. Um dos pressupostos do projecto foi o receptor de imagens ser um Gameboy™, da marca nipónica Nintendo, por se apresentar como uma consola de arquitectura aberta, portátil e com características ideais para uma aplicação deste género tanto ao nível do preço como no seu desempenho. Uma das aplicações possíveis deste projecto poderá ser, por exemplo, a vigilância de bebés.

Dada a vulgarização das redes locais, sem fios, é prudente pensar à priori na versatilidade da aplicação, no que diz respeito ao uso de diferentes tecnologias de transmissão de dados. Assim, a arquitectura do sistema foi pensada, tendo em conta os diferentes sistemas de transmissão sem fios, deixando em aberto a utilização de outras tecnologias de transmissão, existentes no mercado.

Este projecto, pode ser encarado como um óptimo exercício para sedimentar muitos dos conhecimentos adquiridos e desenvolvidos ao longo dos últimos anos. As áreas temáticas de abrangência deste projecto, Electrónica digital e analógica, Arquitectura de Computadores, Interfaces e Periféricos, Sistemas de Comunicações, Processamento digital de Sinal, são alguns exemplos da transversalidade dos conhecimentos aplicados.

2. Arquitectura do Intercomunicador

O intercomunicador foi pensado para um ambiente multi-utilizador, em que cada elemento da rede tem uma identidade própria. A LAN que sustenta as comunicações é composta por dois ou mais elementos, podendo cada um deles ser observado como um nó do sistema. Esta configuração permite estabelecer uma ligação ponto-a-ponto entre duas entidades da rede, e assim a troca de pacotes de dados. Numa configuração simples, o intercomunicador, é um sistema que permite criar a comunicação entre uma "câmara" e um equipamento de "display". A ligação entre ambos é bi-direccional, o que resulta, num sistema interactivo, em relação ao utilizador. Dado este facto, é possível que o utilizador, através do receptor possa controlar os parâmetros da emissão dos dados. Os parâmetros podem ser muito variados, desde, parâmetros de controlo do sensor de imagem, até ao controlo da posição física do emissor com recurso, por exemplo a motores (servos), que controlam a posição da "câmara".

Nesta aplicação, como se trata de um intercomunicador sem fios, a distância entre dois elementos, prende-se apenas com o alcance dos componentes que estão encarregues por manterem a comunicação. Contudo fica-se com a liberdade de, dentro do raio do alcance dos componentes, transportarmos tanto o emissor como o receptor, para onde pretendermos.

O diagrama apresentado na figura seguinte, representa de uma forma simplificada a arquitectura do sistema que se pretende implementar:

Figura 1: Arquitectura do Sistema

2.1. Emissor

Uma Web-Cam, actualmente, é constituída internamente por um sensor de imagem, que pode ser do tipo CCD, ou CMOS. No primeiro caso, é necessário um interface analógico complicado e um circuito de controlo mais complexo. Os sensores de imagem CMOS, pelo contrário, são pensados para interfaces digitais. A tabela seguinte ilustra as diferenças mais acentuadas entre ambos os tipos de sensores.

Target / Items	CMOS Image Sensor	CCD (Charge Coupled Device)
Image Quality	● Larger dynamic range (60 ~ 100dB) ● Lower S/N (>50 dB)	● Large dynamic range (~60 dB) ● Larger S/N (>60 dB)
Consumption Power	● Single supply voltage (3.3 or 5.0V) ● Lower consumption power (<100 mW)	● Multiple supply voltage ● Larger consumption power (>100 mW)
System Requirements	● Low cost MCU is able to handle ● A/D converter is included in the sensor	● ASSP device is required ● External A/D converter is required
Others	● Digital Interface ● Multiple data output modes	● Analog Interface ● Complex control circuits
Total Cost	● Lower ($10 ~ $25)	● Higher (>$25)

Tabela 1: Sensor CMOS Vs CCD

O uso de um sensor de imagem CMOS, simplifica e vai de encontro ao que se pretende para especificar a aplicação, como se pode observar na tabela de cima, as vantagens são enormes, sendo de salientar o baixo consumo de energia, e claro, o custo do sensor.
O sensor de imagem CMOS escolhido é da Mitsubishi (M64282FP), sendo a base de todo o funcionamento do emissor, é a preto e branco, apresenta uma resolução de 128x128 pixeis e uma profundidade de cor de 8 bits, ou seja, é possível representar uma palete de 256 gamas de cinza. Se tivermos em conta que a imagem é colocada num display pequeno, devido à portabilidade do receptor, a resolução é satisfatória.

2.2. Receptor

Para implementar o receptor será necessária uma plataforma que apresente um LCD a funcionar como display, com pelo menos 128x128 pixeis. Como o sistema a implementar é um sistema digital, será necessário para alem do LCD, hardware dedicado para o controlar. Construir um sistema com um LCD e respectivo controlador é uma tarefa que consome tempo na sua implementação. Assim, optou-se por implementar o receptor, com recurso a um equipamento, já existente, e comercializado em qualquer superfície comercial, que venda artigos dedicados aos fanáticos dos videojogos. O equipamento é uma vulgar consola de videojogos, o Gameboy^{TM} Color, que apresenta uma

arquitectura aberta e amigável, indo de encontro às especificações do sistema que se pretende implementar.

O Gameboy™ Color, é uma consola comercializada pela marca nipónica NINTENDO™, portátil e de simples utilização. O interface com o utilizador é composto por quatro botões, um cursor, um ecrã a cores e um altifalante. Som estéreo está disponível no jack dos auscultadores. O hardware interno inclui um derivado do velhinho Z80®, em conjunto com uma RAM. Mais detalhes são apresentados em anexo, através do esquema que apresenta a arquitectura interna, neste caso do Gameboy™ clássico, já que esta informação não está disponível para o Color, o que é irrelevante, já que este se trata de uma evolução do clássico, e a compatibilidade está assegurada.

As especificações técnicas principais do Gameboy™ Color são:

CPU	8-bit Z80-like CPU running at 8.494304MHz
BUSES	8-bit data-BUS, 16-bit address-BUS
RAM	32kB internal, also 8kB adressroom for external RAM
Video RAM	8kB internal
ROM	32kByte reserved in addressroom for external ROM
Sound	4 channels, stereo
Video	Display: Reflective LCD 160x144 dots (physically) Colors: 56 of 32768 Sprites: 40 sprites (8x8 / 8x16)
Com	One serial port with 8kbps
Power	3 Volts, 0.3 Watts, 2 AA Batteries - about 25 hours

Tabela 2: Especificações Técnicas do GameBoy™

O Gameboy™, foi feito para funcionar como consola de jogos, daí apresentar um acesso directo aos barramentos, isto porque cada jogo é colocado num cartucho independente fisicamente da consola, ou seja, a aplicação a desenvolver seguirá o mesmo conceito, e será totalmente independente do Gameboy™. Isto permite que não haja violação da consola, e nenhuma alteração da mesma, em função da aplicação em questão. Assim é possível desde já, viabilizar este projecto em termos comerciais, já que o produto que se está a desenvolver, funcionará como um mero acessório, podendo ser removido do Gameboy™, a qualquer momento, sem apresentar danos para o mesmo.

2.3. Comunicações

Nas comunicações, dado o propósito do projecto é necessário um sistema de transmissão digital sem fios. Existem inúmeros sistemas de transmissão sem fios utilizados em redes domésticas, tais como o protocolo IEEE802.11b (Wi-Fi) ou o protocolo DECT. O primeiro é usado para aplicações de redes de computadores "wireless", sendo o DECT, usado principalmente para transmissão digital de dados de voz, nunca descartando a hipótese de transmitir outro tipo de dados. A solução DECT, foi estudada, e é sem dúvida uma boa opção para o sistema que se pretende desenvolver, o impedimento reside no facto, desta tecnologia não ser a mais barata do mercado.

Optou-se, então por uma solução mais económica, um transceiver digital FM e half-duplex, que pode debitar uma taxa de transmissão máxima de 160kbps, o que para as especificações pretendidas é satisfatório. O módulo em questão é o BiM2-433-160, da Radiometrix, que para além das características já descritas é óptimo para aplicações portáteis, dado o seu baixo consumo de energia. Nestes módulos, o alcance máximo, ou seja a distância máxima entre o emissor e o receptor, é de 200 em campo aberto, e 50m em interiores. Este alcance deve-se às normas impostas para as comunicações de Rádio Frequência, que limitam a potência que se pode radiar. Este transceiver tem uma grande vantagem, permite trabalhar directamente com níveis lógicos compatíveis com TTL (0 a 5V), o que facilita a ligação a qualquer microcontrolador. O protocolo RS232, foi usado na transmissão dos dados, devido à simplicidade de implementação, e ao facto de quase todos os microcontroladores apresentarem já no seu interior, uma USART.

Para estabelecer uma comunicação, fiável e organizada, entre o emissor e o receptor foi criado um protocolo de comunicação, designado por GBDTP (GameBoy Data Transfer Protocol), que será descrito em pormenor mais à frente, neste documento.

3. Projecto de Hardware

A escolha do hardware de suporte ao sistema, foi crucial, para especificar o intercomunicador. Os elementos principais, estão escolhidos e são os apresentados anteriormente. Nesta fase pretende-se fundamentar e satisfazer os requisitos desses elementos. Para facilitar a descrição, e organizar a documentação do projecto de hardware, dividiu-se o emissor e o receptor em blocos. Empiricamente, entendeu-se colocar a imagem e o áudio, em blocos distintos, já que estes podem ser vistos, nesta fase, como blocos completamente independentes. Esta divisão está directamente ligada ao elevado número de componentes de cada um destes blocos. Os outros elementos do intercomunicador, apesar de cada um deles formar um novo bloco, são descritos pelo seu nome real, o que facilita à partida o seguimento desta fase do projecto.

O diagrama seguinte apresenta a divisão em blocos escolhida, estando os restantes elementos chave identificados pelo seu próprio nome:

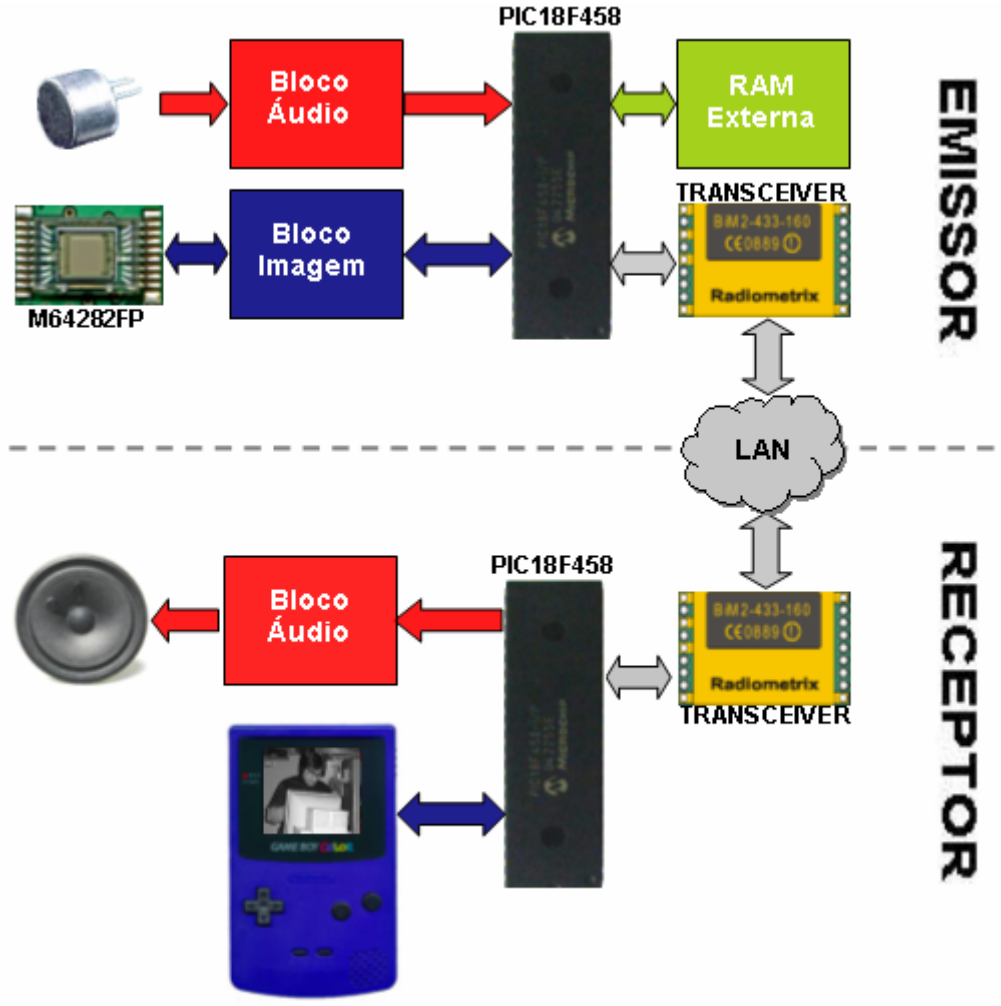


Figura 2: Diagrama de blocos do Intercomunicador

3.1. Emissor

O emissor é baseado no funcionamento da Artificial Retina (AR) da Mitsubishi (M64282FP). Este chip é um sensor de imagem CMOS, de baixo consumo, com resolução de 128x128 pixeis e uma profundidade de cor máxima de 8 bits por pixel (256 tons de cinzento). O sensor, para além de adquirir imagem, tem também capacidade de processamento simultâneo e em tempo-real da imagem que está a adquirir (comportamento similar à retina humana), daí o nome Artificial Retina.

O objectivo do emissor é ter capacidade de ler imagens da Artificial Retina (AR), e enviá-las para o receptor através do módulo de RF. O número de imagens por segundo (frame rate), está directamente relacionada com a taxa de transmissão suportada pelos módulos de RF responsáveis pelas comunicações. Contudo, este sensor permite teoricamente, obter no máximo uma frame rate de ~31 imagens por segundo, o que será impossível nesta aplicação, devido à largura de banda que seria necessária no canal de transmissão.

A Artificial Retina (AR), por si só, não contém nenhum acessório óptico, o que torna impossível a sua aplicação, numa fase de protótipo. Devido a esta razão, foi usada uma Gameboy™ Camera, poupando-nos assim a tarefa de escolher a lente apropriada, e também, a tarefa de soldar os micro pinos do tipo SMD, que compõem a AR.

Devido ao seu funcionamento, a AR, despeja na sua saída analógica uma série de 16,384 valores analógicos (1 valor por pixel, 128x128 pixeis), quando estes dados são lidos da AR, é necessário satisfazer certos requisitos temporais inerentes ao funcionamento da mesma. Cada valor analógico, na saída da AR, tem uma vida muito curta, sendo o tempo disponível para ler cada um desses valores, muito pequeno. Devido a este factor é recomendado pelo fabricante da AR, que xck (sinal de relógio cedido à AR) tenha uma frequência exacta de 500 KHz. Por princípio, foi assumido que os 500 KHz são apenas a frequência máxima de relógio, e que qualquer valor inferior seria aceitável. O pressuposto anterior foi comprovado na prática, e o sinal de xck, acaba por não apresentar uma frequência exacta de 500kHz, mas sim, a frequência possível, mais próxima, que o algoritmo do software e o microcontrolador permitem. É de notar que se diminuirmos demasiado o valor de xck, acabamos por não ter uma imagem instantânea, assim é necessário arranjar um compromisso entre a frame rate pretendida e o xck cedido à AR.

Para amostrar os valores analógicos da saída da AR, apresentados anteriormente, recorremos a uma ADC0820, do tipo Half-Flash, já que a ADC, que o PIC18F458 contém é demasiado lenta. A ADC0820, devido à sua rapidez (tempo de conversão é de 1.5us), não limita o funcionamento da AR, ou seja, é teoricamente possível funcionar com ~31 imagens por segundo.

O controlo do emissor fica a cargo de um PIC18F458 da Microchip, que irá funcionar a 40MHz. Este apresenta portos digitais e analógicos em número suficiente para implementar o emissor, para além das interfaces série, USART, SPI, CAN, e I2C, este último compatível com o módulo DECT estudado, o

HW86010. A escolha deste microcontrolador foi estratégica, já que permite vários tipos de interface para comunicações série, sendo algumas delas amplamente difundidas. A interface série escolhida para implementar o sistema, foi a USART, já que permite comunicar via RS232, através do módulo de RF, com o receptor. No fundo, temos um cabo virtual, a ligar o emissor ao receptor através de RS232!

Para o armazenamento de dados, no emissor, foi adicionada uma SRAM de 8192 Bytes, e um tempo de acesso de 70ns, cuja referência é HY5254A. Isto permite o armazenamento de uma imagem na qualidade máxima sem compressão, ou de pelo menos duas imagens, se for aplicado algum tipo de compressão, como por exemplo o cálculo da diferença de duas imagens consecutivas. O acesso à SRAM é controlado pelo microcontrolador, o qual gera os sinais correspondentes para o barramento de endereços, e os respectivos sinais de controlo da SRAM. O tempo de instrução do PIC18F458 é de 100ns, logo a SRAM escolhida terá de ter um tempo de acesso menor que o respectivo tempo de instrução.

Para a aquisição de áudio, usa-se uma entrada analógica do PIC, programada para o efeito com uma frequência de amostragem de 8kHz, e uma resolução de 8bits por amostra (especificações idênticas às da rede fixa - RDIS) que poderá ser diminuída por software, a quando da transmissão. O sinal sonoro é captado com recurso a um microfone de electreto. O amplificador de áudio é composto por dois andares, no primeiro temos o andar de ganho, e em seguida o andar de filtragem anti-aliasing. Os OP-AMP's utilizados, são do tipo rail-to-rail , já que o circuito está pensado para ser portátil, sendo nesse caso alimentado por baterias, onde apenas estará disponível uma única e positiva tensão de alimentação (5V). Assim foi escolhido o integrado MC602, que apresenta no seu interior dois OP-AMP's do tipo rail-to-rail, um para o andar de ganho, e o outro para implementar o filtro anti-aliasing.

3.1.1. Microcontrolador, PIC18F458

O PIC18F458, da MICROCHIP é o "cérebro" do emissor. A necessidade de controlar os vários periféricos do emissor, e as restrições impostas quanto à velocidade do sistema, levaram à escolha deste modelo. Este dispositivo apresenta-se sob a forma de um chip de 40 pinos, em vários packages, com diferentes dimensões. O package usado, PDIP, é o que apresenta maiores dimensões, dai que seja mais fácil de utilizar na fase de protótipo. Isso não invalida que no futuro, se possam utilizar outros, como por exemplo PLCC, ou mesmo TQFP, com dimensões muito mais reduzidas.

Figura 3: PIC18F458, package PDIP

A escolha deste microcontrolador não foi imediata, numa primeira fase decidiu-se usar uma gama mais baixa da mesma marca, o PIC16F877, que apresenta a mesma configuração física, mas com diferenças em termos de velocidade, nomeadamente da USART, e na dimensão da memória disponível (RAM e ROM menores). A desistência do uso deste microcontolador não invalida a sua aplicação, já que o hardware desenhado, é equivalente, e o software desenvolvido compatível.

As características principais que levaram à escolha do PIC18F458, estão sintetizadas na tabela seguinte:

CPU	8-bit RISC CPU running from DC to 40Mhz
Flash ROM	32 Kbytes
RAM	1536 bytes
EEPROM	256 bytes
I/O PORTS	PORTA → 6 bits PORTB → 8bits PORTC → 8bits PORTD → 8bits PORTE → 3bits } 33 general I/O pins
INTERRUPS	14 interrup sources
TIMERS	3
ADC Module	10-bit, 8-channel Analog-to-Digital converter
Serial Communications	USART@115200bps SSP com SPI e I2C CAN module
Parallel Communications	Parallel Slave Port (PSP) 8-bits wide, with external RD, WR and CS controls
Power	Wide operating voltage range: 2.0V to 5.5V Low-power consumption: 20 µA typical @ 3V, 32 kHz < 1 µA typical standby current

Tabela 3: Características principais do PIC18F458

Como foi apresentado na tabela anterior o PIC18F458, apresenta 5 portos, o que resulta num total de 33 pinos disponíveis para uso geral de entrada/saída digital. Todos os portos podem ainda ser multiplexados com outras funções. As funções que cada porto, executa a dado momento, são programadas por software, podendo um porto ser reconfigurado no decorrer da execução do programa para executar outra tarefa. Como é fácil de perceber, o PIC18F458, torna-se um microcontrolador de certa forma mutante, cuja utilização pode ser moldada, de acordo com as especificações desejadas.

Para desenvolver o hardware do emissor, foi necessário um pormenorizado estudo de todos os periféricos que se pretendiam implementar, já que era

necessário gerir o número de pinos disponíveis. Na figura seguinte está apresentado, o esquema de ligações do PIC18F458, no emissor:

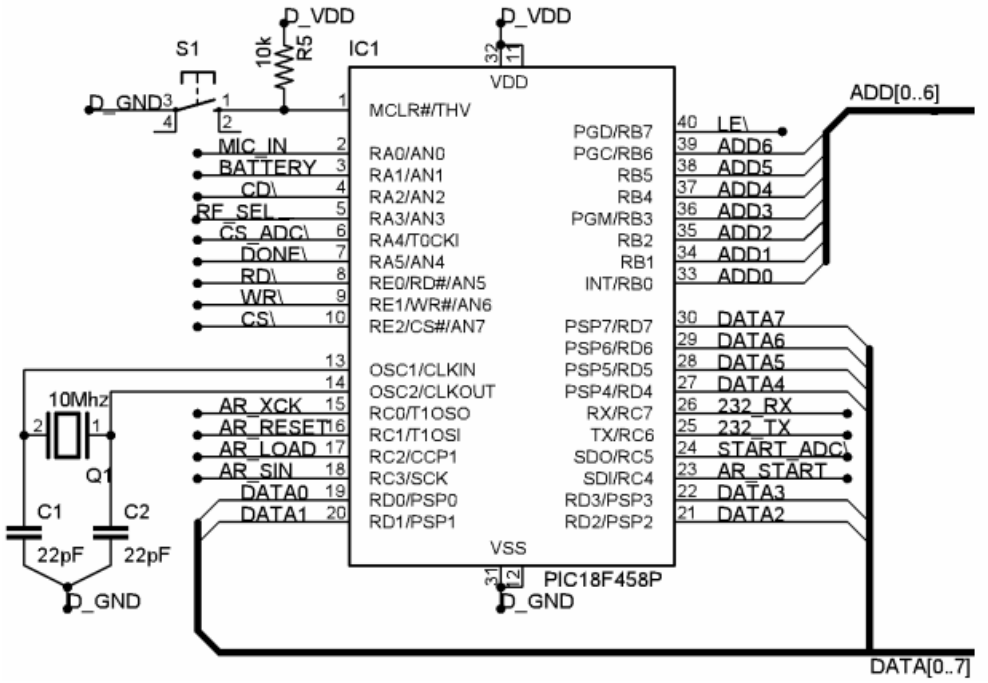


Figura 4: Esquema de ligações ao PIC18F458, no emissor

Para melhor entender as ligações efectuadas no PIC18F458, elaborou-se uma tabela, que sintetiza toda a informação relevante para todos os portos, o domínio do sinal (Analógico/Digital), a respectiva direcção (Entrada/Saída) e claro a função do respectivo pino para a arquitectura proposta.

PORTO	Pino	Nome	A/D	I/O	Função
A	2	AN0	A	I	Áudio
	3	AN1	A	I	Nível de tensão nas baterias
	4	RA2	D	I	Detecção de portadora
	5	RA3	D	O	Selecciona modo de RX/TX do transceiver
	6	RA4	D	O	Chip Select da ADC
	7	RA5	D	I	Fim de conversão da ADC
B	40	RB7	D	O	Latch Enable (373 da RAM Externa)
	39	RB6	D	O	} Barramento da RAM Extrerna
	...	...	...	...	
	33	RB0	D	O	
C	26	RC7	D	I	RX da USART
	25	RC6	D	O	TX da USART
	24	RC5	D	O	Inicia a Conversão da ADC
	23	RC4	D	O	Inicia a Exposição da AR
	18	RC3	D	O	Saída série para programar a AR
	17	RC2	D	O	Programa os registos da AR
	16	RC1	D	O	Reset da AR
	15	RC0	D	O	Clock da AR
D	30	RD7	D	I/O	} Barramento de dados geral
	...	...	...	...	
	19	RD0	D	I/O	
E	10	RE2	D	O	Chip Select da RAM Externa
	9	RE1	D	O	Write na RAM externa
	8	RE0	D	O	Read da RAM externa

Tabela 4: Função, direcção e domínio dos pinos do PIC18F458 do emissor

3.1.2. Bloco de Imagem do Emissor

Como foi dito anteriormente, optou-se por desmantelar uma GameBoy™ Camera, da Nintendo®, dado o facto de esta já incluir a parte óptica, ou seja a lente. Na figura 5, podemos observar o "óculo" que foi usado no emissor. Este "óculo" contém no seu interior, o PCB (figura 6), onde está soldado o sensor M64282FP, uma lente, que é colocada sobre o PCB e uma socket onde temos disponíveis os pinos necessários para o controlo da AR.

Figura 5: Óculo da GameBoy™ Camera

Figura 6: Interior do Óculo, M64282FP

As características principais da AR são:

→ **Resolução,** 128*128 pixeis a preto e branco
→ **Alimentação,** 5.0V single supply
→ **Consumo,** baixo tipicamente 15 mW
→ **Clock,** F_{clock} = 500KHz
→ **Controlo,** de ganho e luminosidade
→ **Sensibilidade,** luz visível e infravermelhos

O sensor apresenta no seu interior 8 registos que têm de ser programados previamente. Estes registos controlam vários parâmetros desde tempos de exposição, ganho, extracção de contrastes, inversão de imagem, calibração, etc.

Assim antes de adquirir uma imagem é necessário calibrar o sensor, através da programação dos seus 8 registos. Esta tarefa é feita através dos pinos dedicados (SIN e LOAD) que o sensor apresenta. Apesar do sensor apresentar 16 pinos, ver figura 7, são apenas utilizados na prática apenas 9 pinos. A representação gráfica apresentada em seguida, contém ambas as numerações. Na figura 7, temos a numeração de todos os pinos do sensor, estando na figura 8, a numeração dos pinos disponíveis no PCB e que serão utilizados, sendo esta, a definição que será seguida ao longo deste documento.

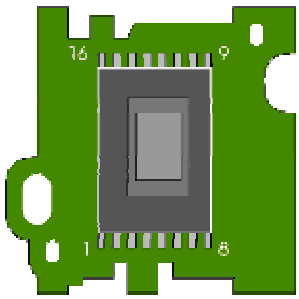


Figura 7: **Frente do PCB que contem a AR**

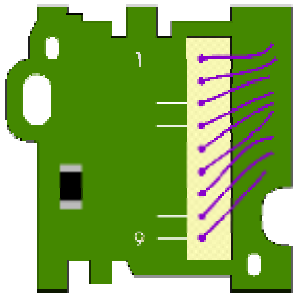


Figura 8: **Verso do PCB que contem a AR**

A descrição e função destes 9 pinos, é feita em seguida, na seguinte tabela:

Pin	Nome	D/A	I/O	Função
1	**Vcc**			Alimentação +5V
2	**Start**	D	I	Inicio de captura de imagem
3	**Sin**	D	I	Entrada série para programar os registos
4	**Load**	D	I	Valida os parâmetros de entrada (Sin)
5	**Reset**	D	I	Reset global do sensor
6	**Xck**	D	I	Entrada de relógio para o MUX
7	**Read**	D	O	Indica quando o sensor está a ler uma imagem
8	**Vout**	A	O	Sinal analógico correspondente à imagem
9	**Gnd**			Massa

Tabela 5: **Pinos disponíveis da AR**

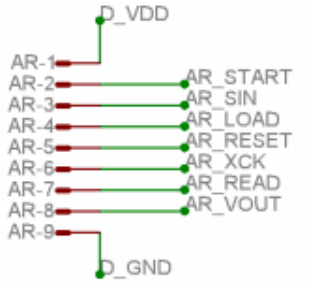


Figura 9: **Esquema de ligações da AR**

O sinal Sin, é uma entrada série de dados, para programação dos 8 registos do sensor. Aqui é necessário enviar em primeiro lugar o endereço do registo (3 bits) e em seguida o valor a programar no registo (8 bits). Finalmente este valor é validado com um sinal de Load.

O diagrama seguinte ilustra a programação de um registo na AR:

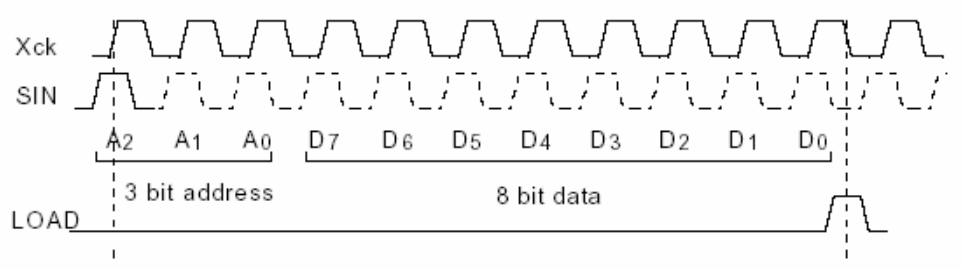


Figura 10: Programação de um registo da AR

Depois de programar todos os registos, tempo de exposição, ganho, etc, o início da captura é iniciado com um sinal de Start. Logo a seguir ao sinal de Start, é necessário esperar que passe o tempo de exposição que se encontra nos registos dedicados que foram programados previamente. Um número de ciclos de relógio correspondentes ao tempo de exposição é fornecido ao sensor pelo PIC. Após esse tempo, o pino Vout começa a debitar valores analógicos, sincronizados com a subida automática do sinal de Read.
O diagrama seguinte, representa o início de captura de uma frame no sensor:

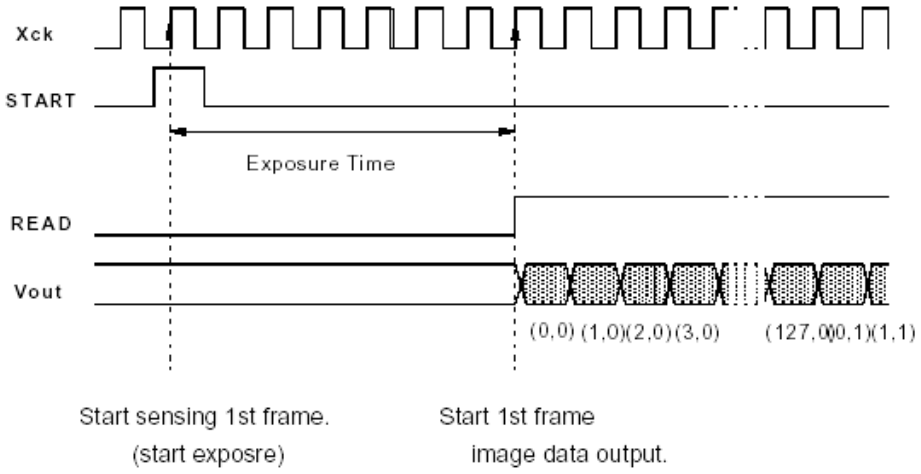


Fig ura11: Inicio da captura de uma imagem

Na tabela 5, estão descritos os pinos, o respectivo domínio do sinal, e a indicação de Entrada/Saída dos mesmos. Temos assim, que todos os sinais são de domínio digital, excepto Vout, que é analógico. Todos são entradas, excepto, Vout e Read, que são saídas. Todos os sinais digitais que são entradas são fornecidos pelo microcontrolador, restando apenas os dois sinais de saída, Vout e Read.

O sinal de Read não será utilizado, já que por software é possível medir o tempo de exposição pré-programado e assim estimar o número de ciclos que é necessário fornecer à AR, para que o tempo de exposição seja cumprido. Neste caso liberta-se um pino digital do microcontrolador, e juntando o útil ao agradável, utiliza-se este sinal como interruptor no circuito de alimentação de led de infravermelhos, ver figura 12, colocando assim o sensor num modo de visão nocturna. Neste caso, o led está apenas aceso quando a frame está a ser lida, o que é óptimo em relação ao consumo de energia, já que cada frame tem

um tempo de amostragem de ~29ms, estando o led de infravermelhos o resto do tempo desligado.

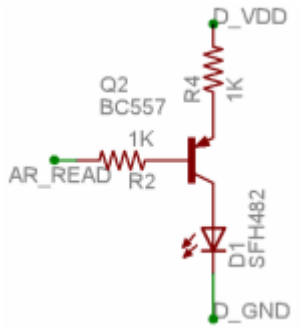


Figura 12: Interruptor do LED de Infravermelhos

O sinal analógico de saída do sensor apresenta uma gama dinâmica de 2Vpp, podendo ser quantizado em 256 níveis. Neste caso estamos na resolução máxima, ou seja 8 bits/pixel. Para a aplicação em questão, apesar do sinal ser amostrado na resolução máxima de 8 bits, no emissor, a resolução será diminuída por software, para que seja possível satisfazer o compromisso, tendo em conta a largura de banda disponível, na transmissão de dados. Assim para a aplicação em questão, ver fig.9, o sinal é quantizado em 16 níveis (ou seja 16 cores), ficando assim com os respectivos 4 bits por pixel. A figura 13, representa quatro instantes de amostragem seguidos, em que o sensor lê uma sequência de pixeis invulgar, ou seja Preto→Branco→Preto→Branco. Este caso na realidade, tem uma probabilidade muito baixa de acontecer durante a captura de uma imagem, por isso esta sequência é usada para o sincronismo de início de imagem. Mais à frente será descrito com mais pormenor, aquando da descrição do protocolo das comunicações.

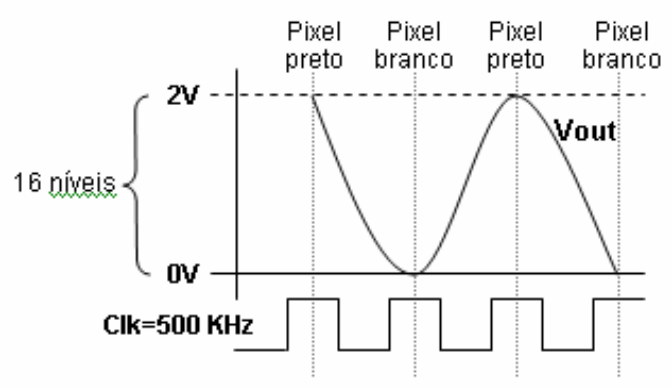


Figura 13: Dinâmica do sinal analógico de saída da AR

Para amostrar, o sinal anterior, recorreu-se a uma ADC externa. A escolhida foi a ADC0820, do tipo half-flash, com uma resolução de 8 bits e um tempo de conversão de 1,5µs. A ADC0820, é feita para interfaces directos com microcontroladores, já que apresenta um pino de Chip Select, o que elimina logo à partida o uso de um buffer, no caso de um interface com o barramento de dados.

Neste caso serão necessários os seguintes sinais digitais de controlo da ADC:

→ CS_ADC\ (CS\), que selecciona a ADC
→ START_ADC\ (RD), que inicia a conversão
→ DONE\ (INT\), sinaliza o fim de conversão

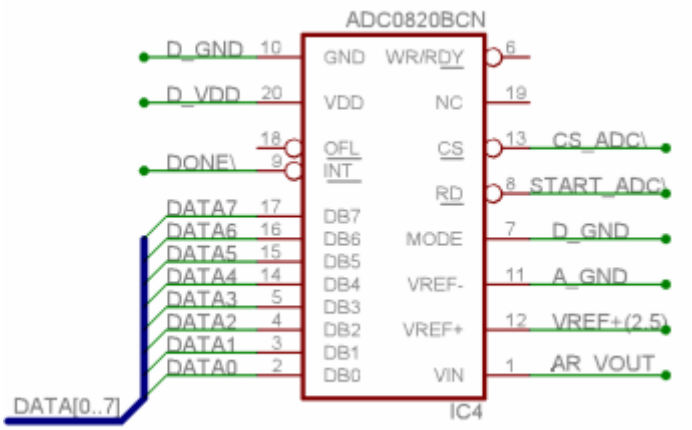


Figura 14: Esquema de ligações da ADC0820 (RD mode)

Esta ADC, pode funcionar em dois modos distintos, o RD Mode, e o WR-RD Mode, tendo sido configurada para esta aplicação, no RD Mode. Neste modo, com o pino MODE, à massa, a ADC0820 funciona no modo de READ. A conversão é iniciada quando o pino RD\ (sinal START_ADC\), vai a zero. Nesta configuração, uma conversão é completada, mantendo o pino RD\ a zero até que os dados apareçam na saída. Quando os dados estão disponíveis uma interrupção é gerada através do pino INT\ (sinal DONE\), de modo a sinalizar o microcontrolador que os dados estão validos e prontos para serem lidos.

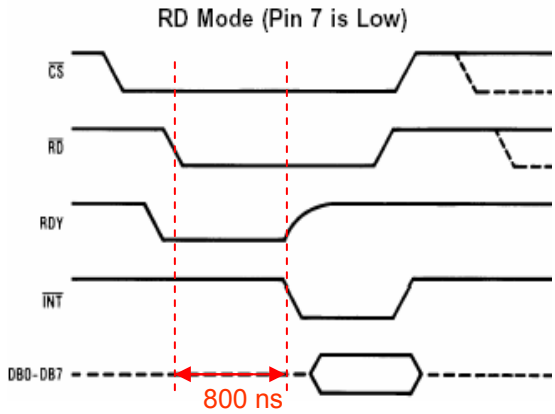


Figura 15: Diagrama de funcionamento da ADC0820(RD Mode)

A solução encontrada passa pelo método acima descrito. Como é necessário ler a um ritmo constante, que representa a resolução máxima de cada imagem, o nosso sinal de START_ADC\ é um sinal, sincronizado pelo microcontrolador, ficando em seguida, o PIC a fazer pooling no sinal INT\ (DONE\) até que haja dados válidos. Na figura 15, está representado o diagrama temporal da ADC0820, neste modo de operação, onde se pode observar que uma amostra está convertida quando estão passados 800ns, desde o inicio da conversão.

3.1.3. Bloco de Áudio do Emissor

O som é adquirido, com recurso a um microfone de electreto. Considerando o esquema da figura 16, tem-se, que no 1º andar o sinal é amplificado, sendo o 2º andar usado para o filtro anti-aliasing. No andar de amplificação tem-se um OP-AMP do MCP602, a funcionar no modo não inversor, cuja tensão de referência, é dada por Vref. Neste caso é importante trabalhar com uma Vref aproximadamente igual a 2.3V, já que estes OP-AMP's não possuem uma saída dinâmica de 0 a 5V, mas sim entre 0 e 4.6V.

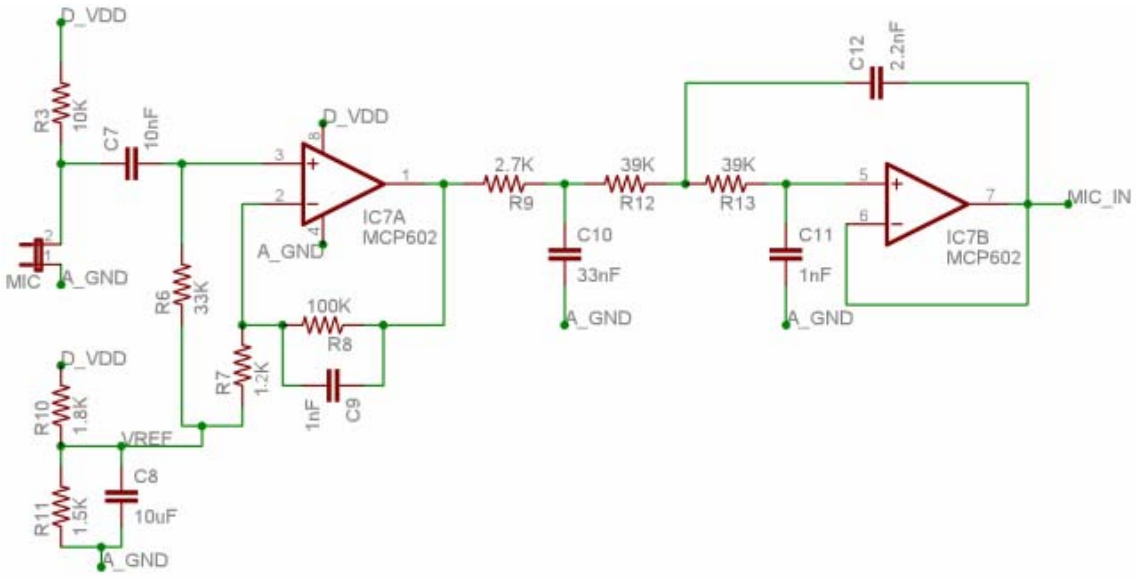


Figura 16: Esquema de ligações do amplificador e filtro de áudio

O ganho, DC Gain = -R8/R7, para R8=100K e R7=1.2K temos que,

DC Gain = - 100/1.2 = -83.33

Neste caso para um sinal de 50mv na entrada do 1º andar tem-se na saída, desse mesmo andar, ~4.2V, o que é satisfatório, já que a tensão máxima de saída do microfone de electreto, testada, ronda os 50mV. A entrada analógica do PIC utilizada, é AN0, sendo programada para uma frequência de amostragem de 8KHz.

Na passagem do domínio analógico para o digital é necessário ter em consideração a frequência de amostragem que se pretende utilizar, para que antes de amostrar o sinal, se eliminem as componentes de frequência que provocam o fenómeno do aliasing. Neste caso pretende-se uma frequência de amostragem de 8KHz, ou seja, pelo teorema da amostragem, temos uma largura de banda desde DC, até aos 4KHz. Na filtragem do sinal, temos um filtro passa-banda no primeiro andar, cujas frequências de corte Fc_L, e Fc_H são calculadas a seguir.

Assim para R6=33K, R7=1,2K, C7=10nF e C9=1nF temos que:

$$Fc_{1L} = (1/2\pi) * \sqrt{(1/(R6.C7))} = 8{,}76Hz$$

$$Fc_{1H} = (1/2\pi) * \sqrt{(1/(R8.C9.R9.C10))} = 1686Hz$$

No 2º andar, é necessário reforçar a filtragem efectuada no primeiro andar para garantir que à frequência de 4KHz, o sinal já está bastante atenuado. O filtro implementado é um passa-baixo de 2ª ordem, numa configuração do tipo Sallen-and-Key, cujas especificações são demonstradas em seguida.

O ganho, neste andar é unitário já que a malha de realimentação é -1, sendo a frequência de corte dada pelos pólos dominantes do circuito representado na figura anterior. Assim para R12=R13=39K e C11=1nF C12=2.2nF temos que:

$$Fc_2 = (1/2\pi) * \sqrt{(1/(R13.R12.C12.C11))} = 2751Hz$$

A resposta em frequência aproximada é dada no gráfico de bode seguinte:

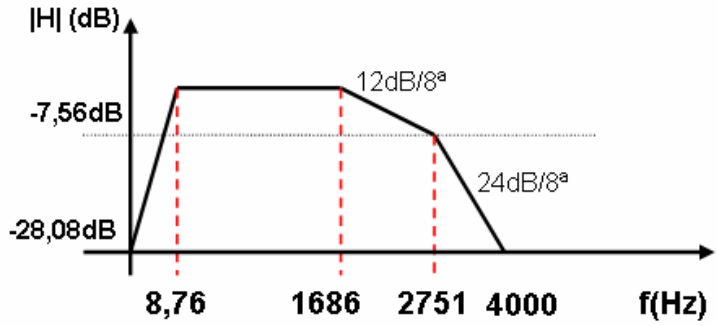


Figura 17: Resposta em frequência do filtro

Como se pode observar na figura 17, o sinal de áudio a 4KHz, já se encontra cerca de 28dB atenuado, o que em termos de energia é bastante, permitindo a amostragem sem problemas quanto ao fenómeno do aliasing.

3.1.4. RAM Externa, 8 KBytes

Para o armazenamento de dados, recorreu-se a uma SRAM de 8KB (8192*8bits), com um tempo de acesso de 70ns. Como o PIC tem um tempo de instrução de 100ns, é teoricamente possível, com esta SRAM, garantir dois acessos à memória externa em duas instruções seguidas. Na prática isto não se verifica, já que o PIC, não possui um BUS de endereços. Então, é necessário multiplexar, um porto genérico de 8 bits, através de uma latch, para escrever um endereço com mais de 8 bits. Neste caso é sempre necessária mais do que uma instrução para aceder a um endereço de memória. A figura seguinte, apresenta a solução encontrada para ligar uma SRAM de 8KB, ao PIC.

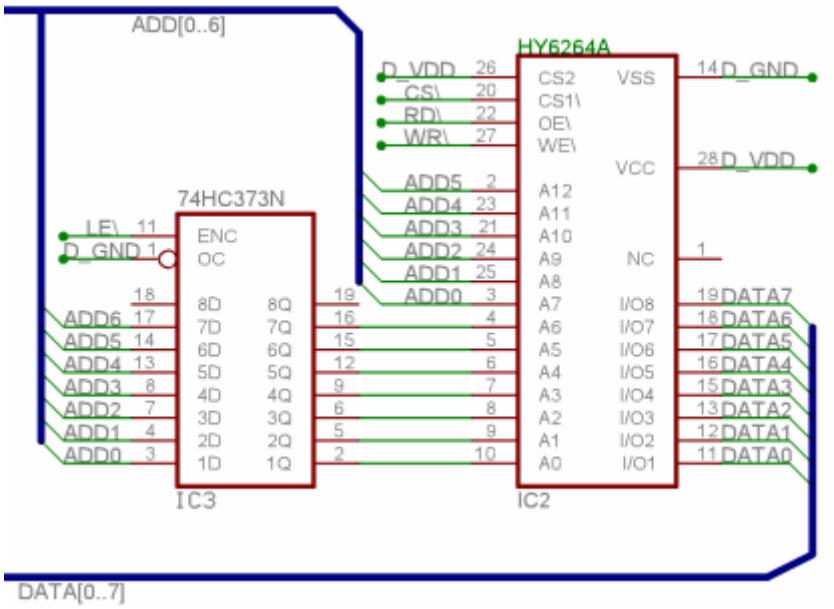


Figura 18: Esquema de ligações da RAM externa

O porto B do PIC, do tipo latched, é escolhido como BUS de endereços, sendo o seu bit mais significativo (RB7), o sinal de Latch Enable (LE\). Nesta configuração, são necessárias duas escritas no PORTB, para aceder a uma posição de memória, sendo possível, endereçar um máximo de 2^{14} Bytes, ou seja 16KBytes. A primeira escrita é feita com a latch activada (LE\ a zero), aqui são colocados os bits menos significativos do endereço na SRAM, em seguida a latch é desactivada (LE\ a um) e são escritos os bits mais significativos na SRAM. Nesta altura o endereço já está válido, sendo em seguida activado o sinal de Chip Select (CS\) da SRAM, e o respectivo sinal de leitura (OE\) ou escrita (WE\). A latch usada, 74HC373, apresenta nas saídas buffers tri-state, que são controlados pelo sinal Output Enable (OE\), estes podem estar sempre "ON", ou seja à massa, porque neste caso não há partilha do barramento por mais dispositivos.

3.1.5. Transceiver no emissor, BiM2-433-160

Para assegurar as comunicações sem fios, foram escolhidos os módulos transceivers **BiM2-433-160** da Radiometrix. Estes módulos permitem a transferência em Half-Duplex, na banda FM, a 433,92 MHz. O BiM2-433-160 tem como mais valia para esta aplicação, um baixo consumo de energia (< 20 mA), e facilidade de interface com a USART do PIC, já que este aceita níveis lógicos entre 0 e 5V, sendo assim dispensável um conversor de níveis de tensão para o protocolo RS232 (MAX232).

Os pinos e os respectivos sinais podem ser vistos na figura seguinte.

Figura 19: **Radiometrix, BiM2-433-160**

O **BiM2-33-160**, está fisicamente dividido, RF do lado esquerdo, dados e controlo do lado direito. Existem dois pinos de controlo (RX select\ TX select\) e um de status (CD\). As combinações possíveis são controladas pelo PIC e podem ser as seguintes:

Complementares

Pin 15 TX	Pin 16 RX	Function
1	1	power down (<1µA)
1	0	receiver enabled
0	1	transmitter enabled
0	0	self test loop back

Figura 20: **Modos de funcionamento do BiM2-433-160**

O módulo RF apresenta quatro modos de funcionamento. Nesta aplicação, o modo de "power down" e o modo de "self test loop back" não são necessários. Como se pode observar na tabela 4, o transceiver está no modo de recepção, quando TX e RX são respectivamente, 1 e 0, e no modo de emissão, quando TX e RX, são respectivamente 0 e 1. Como foi demonstrado anteriormente para os dois diferentes modos de funcionamento, o transceiver apresenta os sinais TX e RX, complementares, sendo assim possível utilizar apenas uma linha para seleccionar o modo de funcionamento do transceiver. Uma solução simples, ver figura seguinte, é inverter o sinal de selecção disponibilizado pelo PIC, através de um inversor feito com um transístor, eliminado assim o uso de uma gate externa.

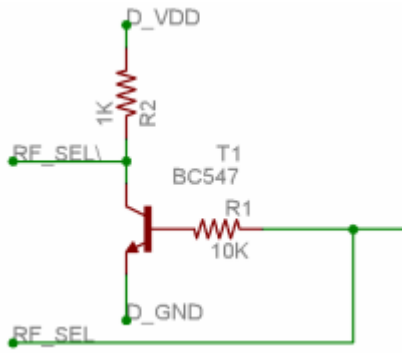


Figura 21: **Esquema do inversor de RF_SEL**

O sinal de detecção de portadora (CD\), é um sinal de status, e só está em funcionamento quando o transceiver está a funcionar como receptor. Nesta situação (modo RX), quando detecta uma portadora, CD\ vai a zero.

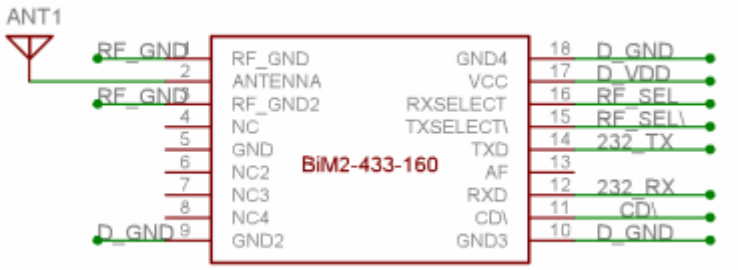


Figura 22: **Esquema de ligações do BiM2-433-160, do emissor**

A ideia é fazer polling em CD\, e perceber quando estamos a receber dados no canal de transmissão. O transceiver do emissor é sempre o slave na transmissão, por isso, irá funcionar num modo intermitente, ou seja, envia uma trama, e logo em seguida fica um tempo pré definido à escuta no canal (fazendo polling a CD\ ), isto para verificar se existe a intenção de enviar um pacote de controlo por parte do receptor. Este método será descrito com mais pormenores aquando da descrição do funcionamento do protocolo.

3.1.6. Driver RS232, MAX232

Para criar uma ligação física entre o emissor e um PC, para futura reprogramação do código, ou por ventura para este funcionar como uma Web-cam, implementou-se um circuito baseado num MAX232, que tem como finalidade, funcionar como driver RS232 standard, apresentado na figura seguinte.

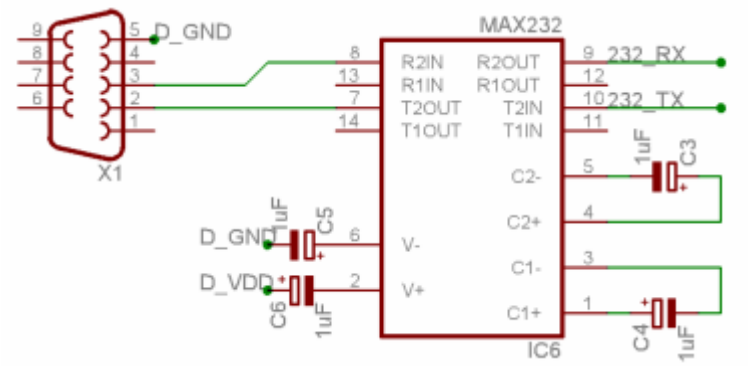


Figura 23: **Circuito do driver de RS-232**

3.1.7. Alimentação do emissor

Para alimentar o emissor, desenvolveu-se um circuito que sugere a utilização de baterias recarregáveis, este circuito é apresentado na figura 24. A bateria a utilizar deve ser de 9V, podendo ser carregada quando o emissor se encontra ligado a uma tensão externa entre 9-12V DC. Quando a bateria se encontra carregada, o díodo de zener (díodo *DZ* da figura apresentada) entra em condução, já que a bateria apresenta aos seus terminais uma tensão superior a

9.6V. Para monitorizar o nível de tensão da bateria, usou-se um pino do PIC18F458, configurado como ADC, para ler regularmente a tensão Vbattery. Esta tensão é metade da tensão aos terminais da bateria, devido ao divisor de tensão aí colocado. Assim a bateria está carregada, quando Vbattery é igual ou superior a metade de 9.6V, ou seja Vbattery > 4,8V. Para garantir um VDD com precisão de 5V, usou-se o regulador de tensão, o MC7805.

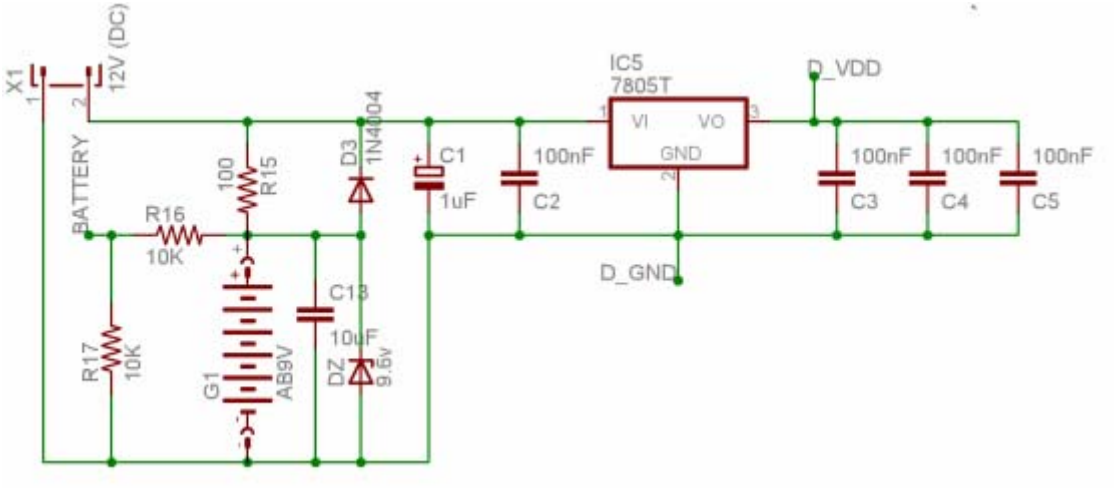


Figura 24: Circuito de alimentação do emissor

3.2. Receptor

No Receptor, como já foi dito anteriormente, foi escolhido o GameBoy™ como consola de visionamento das imagens, sendo o seu display, o destino das imagens vindas do emissor. O GBC, possui uma RAM interna de 32KB, para dados dinâmicos, variáveis, arrays, etc…, mas não possui memória de massa, dedicada a armazenar o código desenvolvido no âmbito do projecto. Assim, para poder desenvolver o protótipo do receptor, construiu-se um cartucho de desenvolvimento.

Para construir o cartucho de desenvolvimento, recorreu-se a um velho jogo, e substituiu-se a ROM, que continha o jogo, por uma ROM do tipo FLASH. A ROM escolhida foi uma AMD-29F040J, que permite com o recurso a um programador, enviar o código desenvolvido para o cartucho e assim testar no hardware.

Figura 25: Cartucho de desenvolvimento

No cartucho de desenvolvimento foi colocada uma ficha de 32 pinos que reproduz exactamente os pinos disponibilizados pelo GB para I/O, ou seja DATA bus (D7-D0), ADDRESS bus (A15-A0), GND, CS\, RD\, WR\, RST\, CLK e Vcc, que será a alimentação do receptor.

Figura 26: X-CHANGER, programador de cartuchos

O controlo do receptor, para garantir compatibilidade de interfaces entre emissor e receptor, fica a cargo do mesmo microcontrolador, o PIC18F458 da Microchip, que irá funcionar, também a 40MHz. Assim garantimos que o sistema possa usar uma das interfaces série disponíveis, (USART, SPI, CAN, e I2C) para estabelecer as comunicações entre dois nós do sistema. Neste caso usou-se também a interface série (USART) para comunicar via RS232, através do módulo de RF, com o emissor.

O interface com o GameBoy™, é feito pelo PIC18F458, através do porto D, que é programado no modo PSP (Parallel Slave Port). Nesta configuração o GameBoy™ pode ler e escrever no PIC18F458, em modo assíncrono e aleatório, comportando-se o PIC18F458 como escravo na leitura e na escrita, podendo ser visto do lado do GameBoy™ como um simples acesso a uma posição de memória.

O áudio, é reposto para o domínio analógico através de uma DAC0832 de interface paralelo com 8 bits de resolução. O porto B do PIC18F458, é usado como barramento de áudio para a DAC. Após a conversão A/D é necessário transformar a saída da DAC, de corrente para tensão, e em seguida filtrar o sinal de áudio, com um filtro passa-baixo, que repõe a banda original do sinal. Para estas tarefas são necessários dois OP-AMP's, um a funcionar como buffer, e o outro como filtro passa-baixo. Foi escolhido o integrado MC602, da Microchip, que apresenta no seu interior dois OP-AMP's do tipo rail-to-rail, ou seja com uma única tensão de alimentação. Nesta altura, o sinal está preparado para ser amplificado. Esta tarefa fica a cargo de um amplificador de áudio, TBA820, de 1,2W de potência, o que é satisfatório, já que se pretende excitar, com o sinal, um altifalante de pequenas dimensões.

3.2.1. Microcontrolador, PIC18F458

Tal como no emissor, o PIC18F458, é o "cérebro" do receptor. A escolha de um microcontrolador igual nos dois nós da comunicação é importante. Isto permite estabelecer uma ligação entre ambos através de uma interface série específica, ou seja, neste caso estamos a utilizar a USART, mas poderia ser utilizada outra, como por exemplo, no caso do módulo DECT estudado (HW86010), que usava uma interface série I2C. No receptor, é também necessário controlar vários periféricos sendo impostas restrições quanto à velocidade do sistema, determinantes na escolha de um microcontrolador idêntico ao do emissor. Neste caso como as dimensões finais do sistema terão de ser muito reduzidas, o package ideal seria o TQFP, já que o receptor terá de ser implementado dentro de um cartucho idêntico ao de um vulgar jogo. Nesta fase, em que se pretende elaborar um protótipo, as dimensões não são um problema, e assim usámos o mesmo package do emissor, o PDIP.

Na figura seguinte está apresentado, o esquema de ligações do PIC18F458, no receptor:

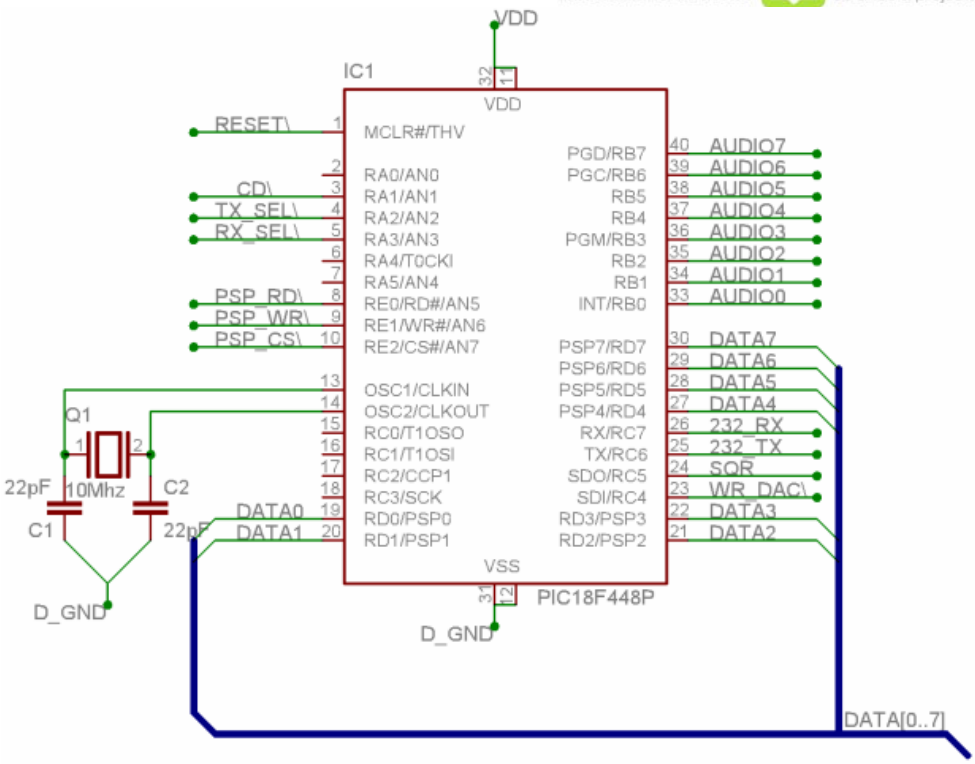


Figura 27: **Esquema de ligações ao PIC18F458, no receptor**

Para melhor entender as ligações efectuadas no PIC18F458, no receptor, elaborou-se uma tabela, que sintetiza toda a informação relevante para todos os portos, o domínio do sinal (Analógico/Digital), a respectiva direcção (Entrada/Saída) e claro a função do respectivo pino para a arquitectura proposta:

PORTO	Pino	Nome	A/D	I/O	Função
A	2	RA0	D	I	NC
	3	RA1	D	I	Detecção de portadora no Transceiver
	4	RA2	D	O	Selector TX, do Transceiver
	5	RA3	D	O	Selector RX, do Transceiver
	6	RA4	D	I	NC
	7	RA5	D	I	NC
B	40	RB7	D	O	Barramento do áudio
	39	RB6	D	O	
	...	...	...	...	
	33	RB0	D	O	
C	26	RC7	D	I	RX da USART
	25	RC6	D	O	TX da USART
	24	RC5	D	O	Gera onda quadrada para Vref negativa
	23	RC4	D	O	Write na DAC
	18	RC3	D	I	NC
	17	RC2	D	I	NC
	16	RC1	D	I	NC
	15	RC0	D	I	NC
D	30	PSP7	D	I/O	Barramento da PSP
	...	...	...	...	
	19	PSP0	D	I/O	
E	10	RE2	D	O	Chip Select da PSP
	9	RE1	D	O	Write na PSP
	8	RE0	D	O	Read da PSP

Tabela 6: **Função, direcção e domínio dos pinos do PIC18F458 do receptor**

No desenvolvimento do software, para o PIC do receptor, numa altura em que a arquitectura já estava decidida, sucedeu-se um problema de relativa importância, já que inviabilizava a comunicação no sentido do GameBoy™ para o PIC18F458. Acontece que o porto D do PIC do receptor está programado para funcionar no modo PSP (Parallel Slave Port). Nesta configuração se for tido em conta o datasheet do PIC18F458, é possível ler e escrever na PSP do PIC, comportando-se este como escravo na transferência de dados. Neste modo de funcionamento, ver figura seguinte, existem duas flags que sinalizam se a operação foi de leitura ou de escrita. Estas flags são IBF (Input Buffer Full), OBF(Output Buffer Full).

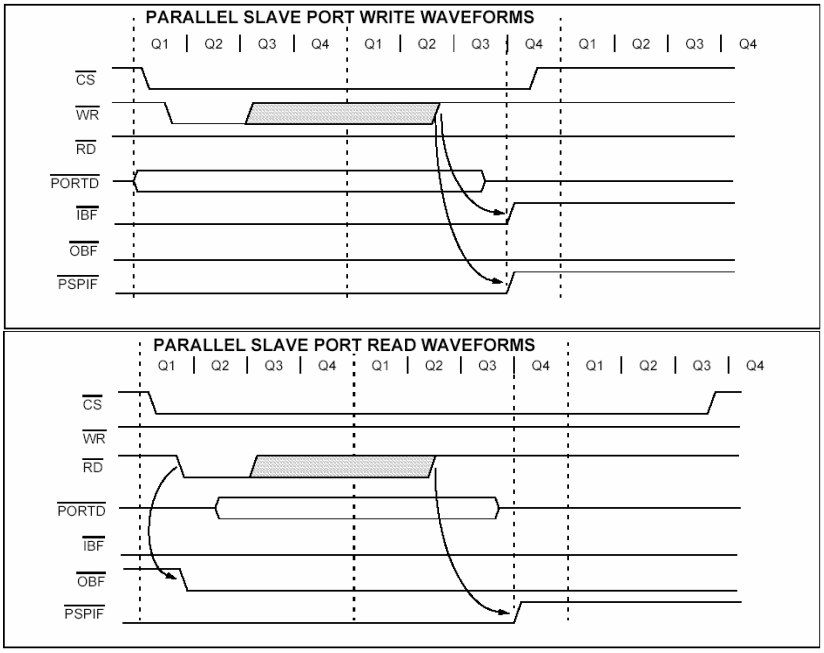


Figura 28: Diagramas de WR e RD, na PSP

Após várias tentativas de comunicar usando as flags IBF e OBF, chegou-se à conclusão que o modo de funcionamento, com o datasheet indica não é possível. Na realidade as flags apresentam um comportamento aleatório, que não dá garantias em relação ao tipo de operação que foi efectuada na PSP.

O bug observado foi constatado no fórum da microchip, ver no site http://forum.microchip.com/tm.asp?m=21301, onde foram encontrados vários textos sobre o problema. Como esta situação foi descoberta, numa fase avançada do projecto, não houve tempo para tentar outras alternativas, procedendo-se apenas à documentação do mesmo.

3.2.2. GameBoy™

O estudo da arquitectura interna do GameBoy™, foi crucial, para especificar, não só o projecto do receptor, mas também o projecto do emissor. A VRAM do GameBoy™ funciona com um sistema de tiles (array de 8 por 8 pixeis) que podem ser guardados de uma forma desordenada, sendo depois ordenados no momento em que se coloca a informação no LCD, através de um mapa de tiles. Este sistema é muito útil quando se pensa em aplicações gráficas repetitivas, caso dos jogos, já que permite, indexar, tiles iguais no mapa e assim poupar espaço na VRAM. A cada tile pode ser atribuída uma palete de 4cores, num universo de 16 paletes. Na realidade o GBC, pode representar imagens de 56 cores de uma palete de 32.768, usa o sistema RGB com 32 níveis de gradiente para cada.

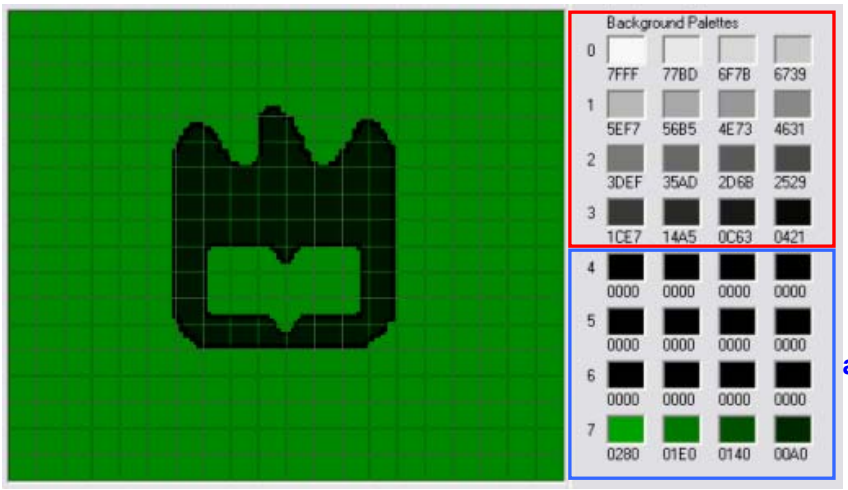


Figura 29: Sistema de tiles e paletes do GameBoy™ Color

Na figura anterior, está apresentada uma imagem, que ocupa todo o LCD do GameBoy™, onde é possível observar a divisão segundo o sistema de tiles acima descrito. A imagem apresenta 160x144 pixeis, o que corresponde a 20x18 tiles. Ao lado estão representadas as paletes de cores correspondentes apenas ao background, estando ainda mais 8 paletes disponíveis. Neste caso serão apenas usadas estas 8 paletes de cor, já que o objectivo principal é ter uma profundidade de cor de 4 bits por pixel, o que resulta num universo de 16 cores, ou mais propriamente, tons de cinza. As primeiras quatro paletes, apresentam o gradiente de cinzentos que ira ser usado para indexar os pixeis de uma imagem, estando as restantes quatro paletes disponíveis para o ambiente gráfico do sistema.

Partindo desta organização da VRAM, é possível comprimir os dados, em termos de paletes de cor e tiles óptimas, no emissor, e em seguida enviá-los para o receptor, já num formato compatível. Um algoritmo que optimize o número de tiles de uma imagem, e extraia da mesma, as suas paletes de cor óptimas, é bem vindo, já que permitiria poupar na largura de banda ocupada na transmissão, e assim aumentar a frame rate. Esta compressão teria forçosamente de ser acompanhada por uma filtragem de *diethering*, que consiste basicamente, na redução da resolução de uma imagem através de um filtro de duas dimensões, sem que haja formação de grandes áreas de cor na

imagem. Exemplos acerca do *dithering,* são apresentados no apêndice B, no final deste documento.

A solução encontrada para o hardware de interface entre o PIC18F458 e o GameBoy™, teve origem no estudo e análise cuidadosa, da organização do mapa de memória do GB.

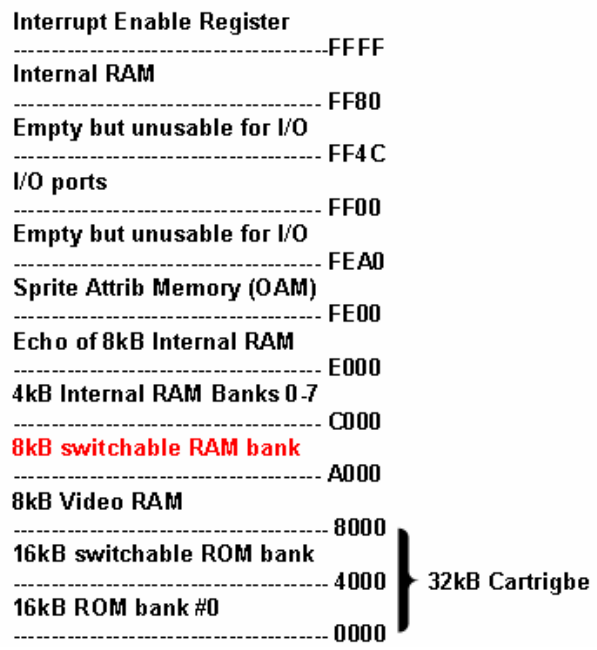


Figura 30: Mapa de memória do GameBoy™ Color

A figura anterior, representa o espaço de endereçamento disponível no GameBoy™ Color, onde se destaca uma zona de extrema importância na solução encontrada. A zona assinalada a vermelho corresponde a um espaço disponível para colocação de uma possível RAM externa. Normalmente, num cartucho de um jogo, esta zona é utilizada para endereçar uma RAM externa, que associada a uma pilha, funciona como uma memória de massa, ou seja, é destinada a guardar dados vitalícios como, por exemplo, o High-Score de um jogo. Neste caso este espaço não é utilizado, sendo possível fazer nesta zona, I/O mapeado em memória. Para isso é necessário recorrer a lógica adicional, que fará a descodificação do respectivo endereço.

Depois de efectuar alguns testes de I/O, no GameBoy™, foi observado um fenómeno de grande importância, que inviabilizava a solução proposta anteriormente. Este fenómeno, que pode ser observado no diagrama da figura seguinte, consiste no facto de que o GB, para múltiplas leituras seguidas num endereço, o sinal respectivo de RD\, fica eternamente a LOW. O problema que esta situação implica é grave já que o PIC18F458, no modo de PSP, necessita de uma transição de High to Low para gerar uma interrupção que sinaliza a intenção, por parte do GB, de leitura de dados. A solução para este problema consistiu em fazer um OR lógico do respectivo sinal de RD\, com o sinal de CLK, disponibilizado pelo GB. Assim gerou-se um novo sinal, PSP RD\, que agora é forçado pelo sinal de CLK, a subir ao estado de HIGH, e assim permitir gerar a interrupção necessária, aquando da descida de HIGH→LOW, para a sinalização do PIC.

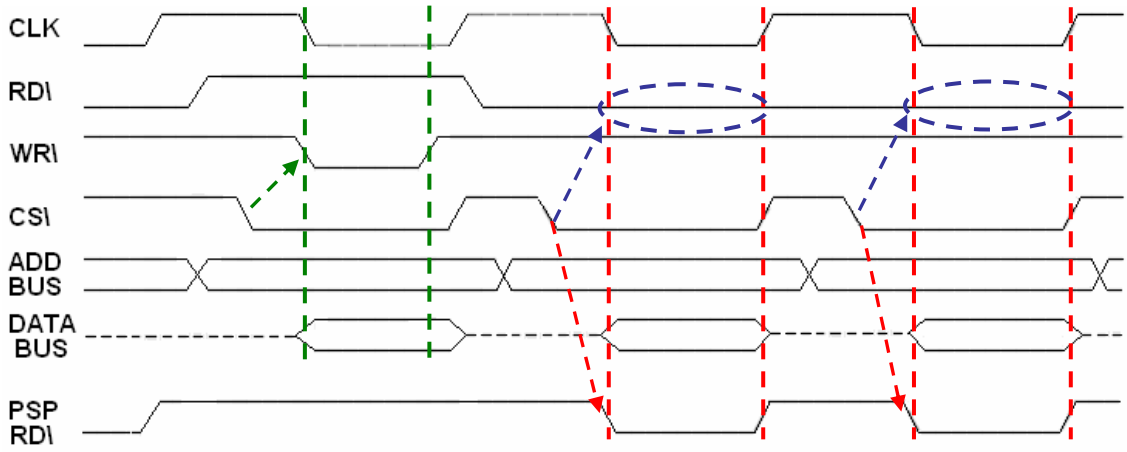


Figura 31: Diagrama de WR e RD do GameBoy™

Na figura seguinte, está representada a lógica necessária para obter os sinais de PSP CS\ e PSP RD\, respectivamente. O sinal de PSP CS\, é gerado através de descodificação de endereços e mapeado na zona de memória reservada pelo GB para uma RAM externa, como foi dito anteriormente, estando esta zona compreendida entre os endereços A000 e BFFF.

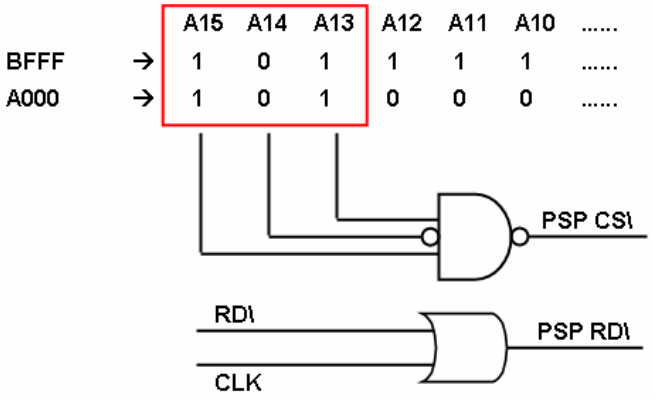


Figura 32: Lógica para gerar os sinais PSP CS\ e PSP RD

Na prática foi usado um chip, que contêm três NAND's, o 74HC10, para descodificar os endereços (ver figura 33), tendo o OR sido gerado com dois díodos de sinal (1N4148), eliminando assim o uso de uma gate.

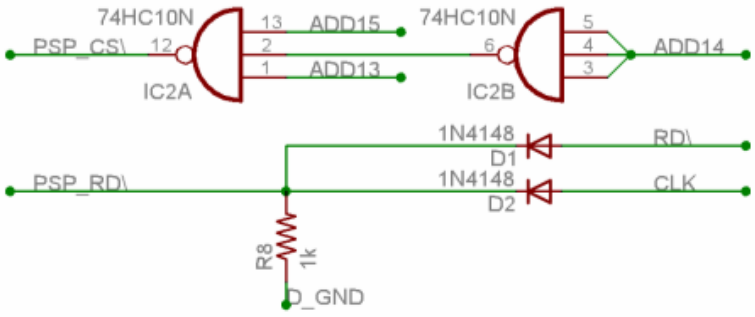


Figura 33: Circuito que gera os sinais de PSP CS\ e PSP RD

3.2.3. Bloco Áudio do receptor

O som é reposto do domínio digital para o analógico, com recurso a uma DAC0832 que apresenta uma resolução de 8 bits e um tempo de conversão de 1µs. Esta DAC é óptima para interface com microcontroladores, já que apresenta linhas de selecção dedicadas para ligação a barramentos de dados gerais.

A DAC0832 necessita de uma tensão de referência, negativa, o que nesta fase é um problema, já que no receptor só temos a tensão de alimentação do circuito disponibilizada pelo GameBoy™, que é de 5V. Para criar uma tensão de referência negativa para a DAC, usou-se uma linha digital do PIC18F458, para gerar uma onda quadrada, que por sua vez será transformada numa tensão negativa através de um circuito do tipo charge-pump.

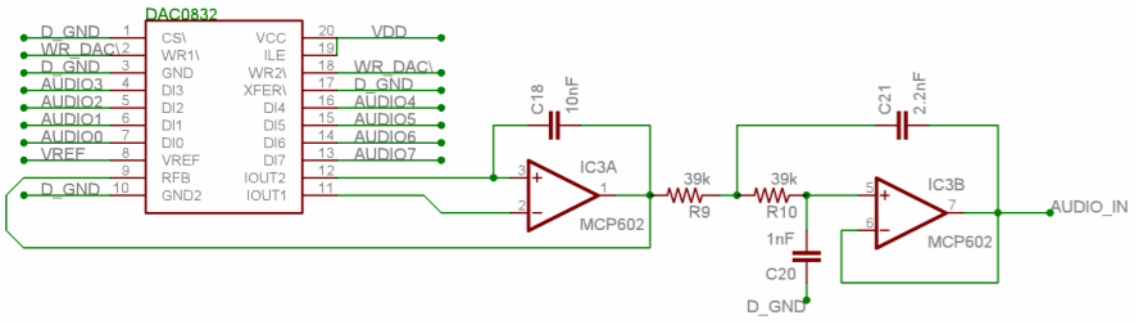


Figura 34: Circuito de condicionamento e filtragem do sinal de áudio

Como a DAC, tem o seu próprio barramento de dados, não é necessário utilizar o CS\ para a seleccionar aquando da escrita. Os pinos de Input Latch Enable (ILE) e de Transfer Control Signal (XFER), não são utilizados no modo single-buffer, daí que estejam eternamente ligados a Vcc e GND, respectivamente. O pino de Vref, que correspondente à tensão de referência negativa, é alimentado pelo circuito da figura seguinte, estando o valor de Vref directamente relacionado com a amplitude do sinal de saída. Dada esta situação, é possível controlar o volume do som, através do controlo por software do duty-cycle da onda quadrada.

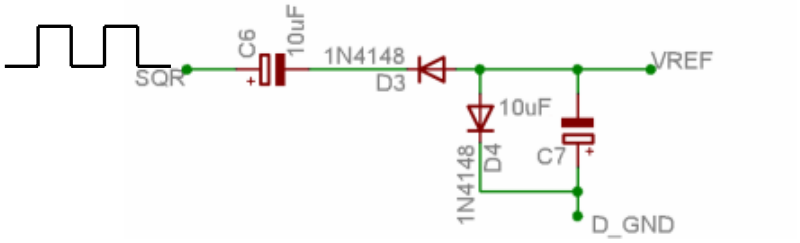


Figura 35: Circuito que gera uma tensão de referência negativa

Na sequência do circuito da DAC0832, temos os dois OP-AMP's, encapsulados num chip, o MCP602, rail-to-rail, onde o primeiro funciona como buffer que permite converter a corrente de saída da DAC numa tensão correspondente, e o segundo como um filtro passa-baixo. As especificações do filtro são idênticas às do filtro anti-aliasing implementado no emissor, já que o sinal será reposto para o domínio analógico, também a uma frequência de conversão de 8KHz,

ou seja, pelo teorema da amostragem, temos uma largura de banda desde DC, até aos 4KHz. O filtro passa-baixo tem como função repôr a banda que o sinal tinha antes de ser amostrado, e assim eliminar o ruído de quantificação colocado pela DAC. O filtro passa-baixos é de 2ª ordem, com uma configuração do tipo Sallen-and-Key, e tem as seguintes especificações:

→ Ganho unitário
→ Frequência de corte é dada pelos pólos dominantes.
Com R9=R10=39K e C20=1nF C21=2.2nF temos e:

$$F_C = (1/2\pi) * \sqrt{(1/(R9.R10.C20.C21))} = 2751Hz$$

Após a filtragem o sinal de áudio está pronto para ser amplificado. Para executar esta tarefa, foi escolhido o amplificador de áudio, TBA820, de 1.2W de potência. O circuito implementado foi o recomendado pela datasheet do fabricante, tendo sido apenas alterada a resistência de feedback (R5=1K), para aumentar o ganho do amplificador para 5, isto para o caso de estar a atacar um altifalante de 8Ω.

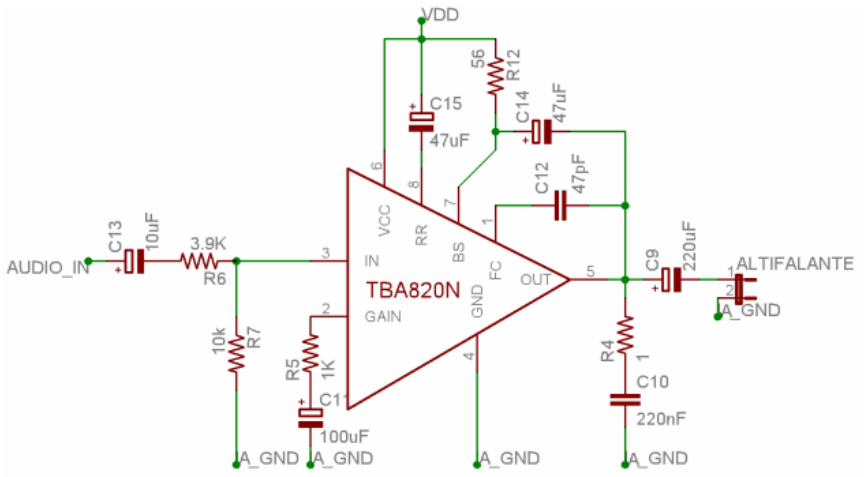


Figura 36: Circuito do amplificador de potência do áudio

3.2.4. Transceiver no receptor, BiM2-433-160

O módulo RF apresenta quatro modos de funcionamento. Nesta aplicação, o modo de "power down" e o modo de "self test loop back" não são necessários. Como se pode observar na figura 20, o transceiver está no modo de recepção, quando TX e RX são respectivamente, 1 e 0, e no modo de emissão, quando TX e RX, são respectivamente 0 e 1. Como no receptor existem pinos digitais em numero suficiente optou-se por utilizar dois diferentes pinos para controlar o modo de funcionamento do transceiver.

O sinal de detecção de portadora (CD\), é um sinal de status, e só está em funcionamento quando o transceiver está a funcionar como receptor. Nesta situação (modo RX), quando detecta uma portadora, CD\ vai a zero.

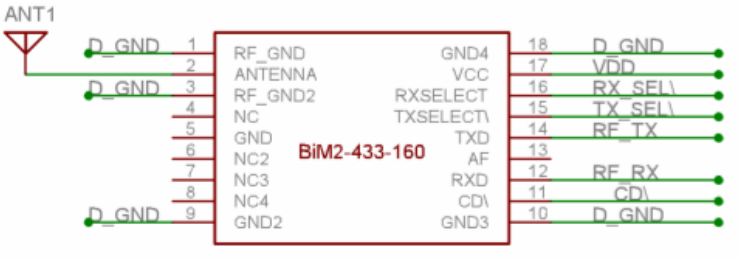


Figura 37: Esquema de ligações do BiM2-433-160, do receptor

Neste caso o transceiver do receptor é o master na transmissão, por isso, funciona no modo que o utilizador define. Este método será descrito com mais pormenores aquando da descrição do funcionamento do protocolo.

4. Protocolos das comunicações

Para estabelecer uma comunicação entre dois nós da LAN, é necessário que exista uma organização dos dados. Esta organização, deve satisfazer vários critérios, tais como, formatação dos dados, sincronismo, identificação e finalmente correcção de eventuais erros. Para solucionar este problema foi implementado um protocolo para a transferência de dados, ao qual foi chamado **GBDTP** (**G**ame**B**oy **D**ata **T**ransfer **P**rotocol). O GBDTP, permite a troca multi-utilizador, de pacotes de dados, de um modo organizado e seguro. Em termos protocolares temos o GBDTP, ao nível da camada da aplicação, estando o protocolo RS232, no fundo a garantir sincronismo na transferência das palavras.

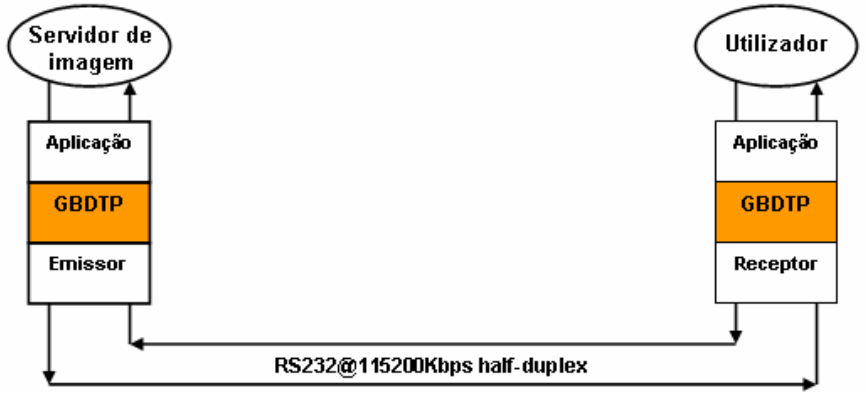


Figura 38: **Camadas protocolares usadas nas comunicações**

4.1. O standard RS232

O protocolo RS232, ainda é um dos mais utilizados no, que diz respeito a comunicações série, devido à sua simplicidade de implementação. O código usado pelo protocolo RS232, é do tipo NRZ (Non Return to Zero). Este código apresenta um problema, já que para uma longa sequência de uns ou zeros, deixa de existir sincronismo de bit, o que dificulta na decisão de um símbolo. No caso de uma comunicação assíncrona, os elementos que se encontram a comunicar, apresentam relógios com frequências idênticas, mas não iguais. Neste caso é óbvio que é impossível comunicar de um modo assíncrono através de um código NRZ puro. Surge então o protocolo RS232, que mistura código NRZ, com bits de sincronismo. Para isso inclui um start bit, oito data bits em NRZ, e um stop bit. Um bit de paridade, e um stop bit adicional podem ser também incluídos. O diagrama seguinte ilustra o protocolo série RS232.

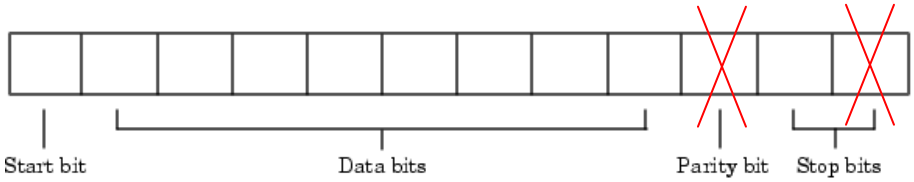


Figura 39: **Protocolo RS232**

O número de bits transferidos por segundo é dado pela baud rate, que inclui todos os bits, de sincronismo, dados e paridade. No modo assíncrono, um byte transmitido deve ser identificado pelos start e stop bits, em que o start bit indica quando o byte de dados está a começar, e o stop bit indica que o byte de dados já foi transferido. A configuração usada no protocolo RS232 é a seguinte:

Baud Rate	:	57600 ou 115200 bps
Data Bits	:	8 bits
Paridade	:	None (ou seja desactivada)
Stop Bits	:	1 bit

Comparando as configurações anteriores com a estrutura protocolar representada na figura anterior, temos que para cada Byte de dados efectivo existe um overhead de 2 bits, sendo um, o start bit, e o outro, o stop bit. Nesta configuração para uma Baud Rate de 115200bps, temos na realidade uma transmissão de dados efectivos igual a:

TxReal = BaudRate*0.8 = 115200*0.8
TxReal = 92160 bps

Para o caso, em que a Baud Rate é 57600bps, temos na realidade uma transmissão de dados efectivos igual a:

TxReal = BaudRate*0.8 = 57600*0.8
TxReal = 46080 bps

4.2. GameBoy Data Transfer Protocol (GBDTP)

Este protocolo foi elaborado a pensar nesta aplicação específica, mas não descarta outras aplicações futuras. Como todos os protocolos, este visa a identificação dos dados, formatação e eliminação de falhas de comunicação. Assim, foram criados, um campo de sincronismo de trama, um de Check Sum, vital para a detecção erros no Header do GBDTP. Para aplicações multi-utilizador existem, um identificador de câmara, e um identificador de GB, tendo sido implementado, também, um campo que identifica o comprimento de bytes referentes aos dados do som e outro que identifica o comprimento relativo aos dados da imagem. O tipo de trama é distinguido num campo dedicado, que identifica se é uma trama de configuração, ou se é uma trama de dados.

O **GBDTP**, apresenta os seguintes campos:

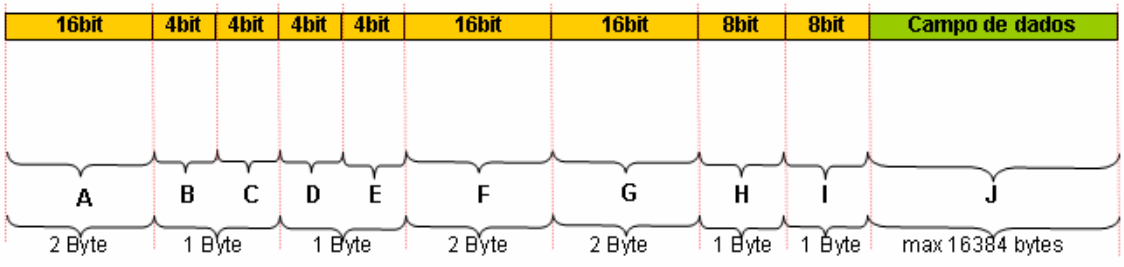


Figura 40: Cabeçalho do GameBoy Data Transfer Protocol (GBDTP)

Ao todo, o GBDTP, apresenta 10 campos, dos quais 9 são o cabeçalho. O cabeçalho apresenta 10 bytes de comprimento. Em seguida, são descritos todos os campos que compõem o cabeçalho:

- **Campo A,** sincronismo de trama, é dado pela sequência inalterável, **1111 0000 1111 0000**, que significa uma sequência de pixeis, preto-branco-preto-branco.

- **Campo B,** identificador da Câmara, permite utilizar, no futuro uma rede com um máximo de 16 câmaras.

- **Campo C,** identificador do GameBoy™, permite utilizar, no futuro uma rede com um máximo de 16 receptores.

- **Campo D,** identificador do tipo de trama, permite decidir qual o tipo de dados da trama:

  0000 – trama de dados de imagem
  0001 – trama de dados de som
  0010 – trama de dados de imagem + som
  0011 – trama de controlo
  …….
  ……. } 12 comandos não usados
  1111

- **Campo E,** versão do GBDTP, indica qual é a versão do protocolo que está a ser usada.

- **Campo F,** indica o comprimento do campo de dados relativos ao som em bytes. Tem 16 bits para permitir enviar um segundo de som, ou seja 2^{16}=8192bytes.

- **Campo G,** indica o comprimento do campo de dados relativos à imagem em bytes. Tem 16 bits para permitir transmitir uma imagem com a máxima resolução.

- **Campo H,** byte livre, para futuras aplicações.

- **Campo I,** byte de Check Sum, é o somatório de todos os bytes que compõe o cabeçalho do GBDTP. Permite detectar se existem dados corrompidos no cabeçalho.

- **Campo J,** campo de dados, em primeiro lugar vêm os dados relativos ao som, e em seguida os dados relativos à imagem, cada uma das componentes tem um máximo de 8192 Bytes.

4.3. Comunicação entre os transceivers

Na realidade, devido ao facto de se estarem a utilizar módulos de RF, do tipo half-duplex, é necessária alguma organização nas comunicações do sistema. Cada módulo de RF, numa dada instância temporal, encontra-se num modo de transmissão, ou a emitir ou a receber dados. Partindo do princípio que o transceiver do receptor é sempre o master na comunicação, se o utilizador, quiser enviar um comando para a câmara, tem de alterar o modo de funcionamento do transceiver de RX para TX, o que sem aviso prévio irá causar uma colisão de dados no canal de transmissão. Esta colisão só existe enquanto o emissor estiver a transmitir, e acaba enquanto este ficar à escuta.

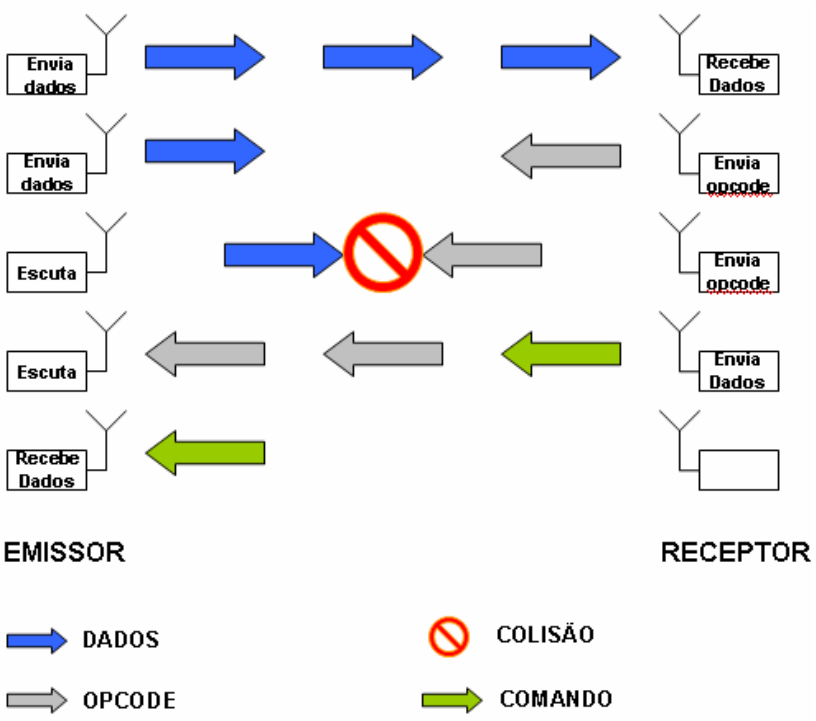


Figura 41: Comunicação entre os transceivers

Para garantir que o comando não se perde na colisão, temos que estimar um tempo mínimo para que a ligação esteja estabelecida e só depois enviar o pacote correspondente ao comando. Ou seja, podemos por exemplo enviar um opcode pré estabelecido, durante um tempo superior a duas escutas consecutivas, do canal, por parte do emissor, e só depois de termos a certeza que se estabeleceu a ligação enviamos o pacote de controlo. Nesta configuração, ver figura anterior, podemos dizer que os transceivers, estão a funcionar num modo intermitente, estando a LAN limitada pelo facto de não existir multiplexagem temporal, ou em frequência, e assim permitir estabelecer várias ligações ponto-a-ponto.

5. Projecto de Software

A estrutura do software desenvolvida no âmbito deste projecto foi dividida em várias etapas. Na primeira etapa, como consequência do estudo da arquitectura interna do GameBoy™, foi desenvolvido e implementado o código a pensar na aplicação partindo do receptor. O estudo da arquitectura interna, fez com que as especificações do intercomunicador fossem de encontro às características do receptor. No desenvolvimento do código para o GameBoy™, foi usado o compilador GBDK (GameBoy Development Kit) para DOS, cuja distribuição é gratuita. O compilador é baseado em linguagem C, contendo várias bibliotecas implicitamente dedicadas ao hardware GameBoy™. O código foi testado num simulador, ver figura seguinte, existente para o Windows®, o NO$GBM, que permite executar o código, com facilidades para o debugging.

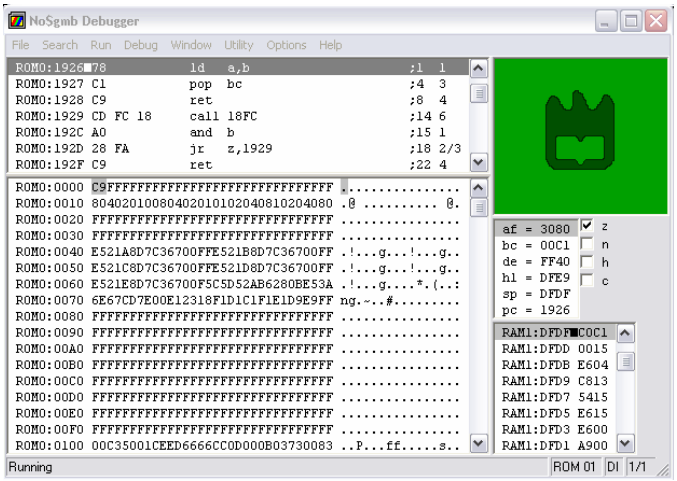


Figura 42: No$GMB, simulador do GameBoy™ para o Windows

Em seguida foi desenvolvido o software relativo aos microcontroladores, tendo por base o compilador PICC-18 da HIGH-TECH® SOFTWARE, que pode ser utilizado com um editor e simulador de software, de ambiente gráfico, o HI-TIDE. Este simulador permite compilar o código escrito e corrê-lo passo a passo, o que facilita o debugging e permite ao programador, através da simulação, entender e perceber como se comporta o hardware.

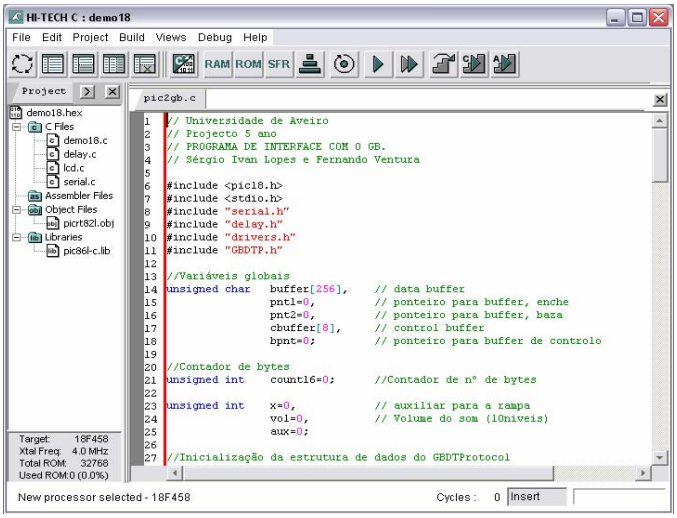


Figura 43: Hi-Tide com PICC-18, editor, simulador e compilador

Por fim, desenvolveu-se uma aplicação gráfica que pudesse ser utilizada como consola de visionamento de imagem, num vulgar PC. O código desta aplicação foi escrito em Visual Basic®, sendo a aplicação pensada para correr nos sistemas operativos Windows 95/NT/98/2000/XP, da Microsoft.

5.1. Estruturas de Dados

A Artificial Retina, contém oito registos internos de 8 bits, que se encontram divididos como mostra a tabela 7. Segundo esta divisão podem considerar-se um total de 14 registos internos, que é necessário programar. Decidiu-se agrupar, os dados relativos aos registos numa estrutura, permitindo assim ter associado a um registo, o endereço, o valor actual, o valor máximo e o seu nome.

Reg. No.	Address	7	6	5	4	3	2	1	0
1	001	N	VH1	VH0	G4	G3	G2	G1	G0
2	010	C17	C16	C15	C14	C13	C12	C11	C10
3	011	C07	C06	C05	C04	C03	C02	C01	C00
4	100	P7	P6	P5	P4	P3	P2	P1	P0
5	101	M7	M6	M5	M4	M3	M2	M1	M0
6	110	X7	X6	X5	X4	X3	X2	X1	X0
7	111	E3	E2	E1	E0	I	V2	V1	V0
0	000	Z1	Z0	O5	O4	O3	O2	O1	O0

Tabela 7: Tabela dos registos internos da Artificial Retina

A estrutura foi defenida como o tipo AR :

```
typedef struct
        {
        unsigned char   bitmask;        // Mascara ao registo
        unsigned char   offset;         // Offset relativo ao registo de 8 bits
        unsigned char   value;          // Valor inicial
        unsigned char   max;            // Valor máximo permitido
        char            key;            // Tecla de interface
        char            name[6];        // Nome do registo
        }AR;
```

Finalmente, foi criado um array de estruturas, do tipo AR, que representam todos os registos, da Artificial Retina, sendo inicializada com os seguintes valores:

```
AR ARreg[14]={  /*      bitmask         offset  value   max     key     name    */
                {       0xC0,           6,      0x02,   0x03,   'z',    " Z "   },
                {       0x3F,           0,      0x1F,   0x3F,   'o',    " O "   },
                {       0x80,           7,      0x00,   0x01,   'n',    " N "   },
                {       0x60,           5,      0x00,   0x03,   'a',    " VH "  },
                {       0x1F,           0,      0x04,   0x1F,   'g',    " G "   },
                {       0xFF,           0,      0x03,   0x0F,   'h',    " C1 "  },
                {       0xFF,           0,      0x00,   0xFF,   'l',    " C0 "  },
                {       0xFF,           0,      0x01,   0xFF,   'p',    " P "   },
                {       0xFF,           0,      0x00,   0xFF,   'm',    " M "   },
                {       0xFF,           0,      0x01,   0xFF,   'x',    " X "   },
```

```
{       0x80,       7,      0x00,   0x01,   '3',    " E3 "  },
{       0x70,       4,      0x00,   0x07,   'e',    " E "   },
{       0x08,       3,      0x00,   0x01,   'i',    " I "   },
{       0x07,       0,      0x01,   0x07,   'v',    " V "   },
};
```

Na implementação do protocolo das comunicações, foi essencial definir uma estrutura de dados dedicada à manipulação dos dados relativos ao GBDTProtocol. A estrutura que define o protocolo, é apresentada em seguida e visa o agrupamento dos dados relativos a uma frame do protocolo na mesma estrutura.

```
typedef struct
        {                               //Campo
        unsigned char sync_1;     // A1  Byte de sincronismo 1
        unsigned char sync_2;     // A2  Byte de sincronismo 2
        unsigned char cam;              // B    Identificador de camera (4bits)
        unsigned char gb;               // C    Identificador de GB (4bits)
        unsigned char type;             // D    Tipo de frame (4bits)
        unsigned char vers;             // E    Versão do GBDTP (4bits)
        unsigned int audio_l;           // F    Comprimento do campo de dados do audio
        unsigned int image_l;           // G    Comprimento do campo de dados do video
        unsigned char free;             // H    Byte livre
        unsigned char cheksum;    // I     Byte de Checksum, controlo de erros
    }GBDTP
```

Para melhor entender esta estrutura de dados é aconselhável consultar o ponto 4.2, deste documento. Neste ponto está especificada a definição do GBDTProtocol, com detalhe, facilitando assim o entendimento desta estrutura de dados.

5.2. Áudio Interrupt

O som é adquirido no emissor a uma frequência de amostragem de 8KHz, sendo depois reposto à mesma frequência no receptor. Em ambos, programa-se o TIMER0, para gerar uma Rotina de Serviço à Interrupção a essa frequência. A RSI, que processa uma amostra de áudio é gerada pela interrupção do overflow do TIMER0. A figura seguinte, ilustra como foi programado o TIMER0 para obter a frequência de amostragem de 8KHz:

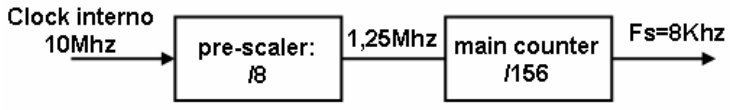


Figura 44: Programação do TIMER0

Iniciou-se o TIMER0, como um contador de 8 bits, em que a fonte de relógio é o clock interno do PIC, com um prescaler de 8. Nestas circunstâncias, o valor que deve ser contado pelo TIMER0, antes de ocorrer uma interrupção é 156. O TIMER0, gera uma interrupt, sempre que chega a 255, então é necessário iniciar o TIMER0, com o valor complementar de 156, ou seja o registo de contagem do timer, TMR0L deve ser inicializado sempre com 99 (255-156=99).

5.3. Bibliotecas Implementadas

A estrutura seguida no projecto de software permitiu desenvolver bibliotecas globais e locais. Bibliotecas Globais, são bibliotecas que poderão ser utilizadas em ambos os nós do sistema, permitindo assim a reutilização do código e optimização do software. As Bibliotecas Locais, foram desenvolvidas para um nó em específico, sendo utilizadas apenas nesse nó. A figura seguinte representa as bibliotecas implementadas e adaptadas no desenvolvimento do software.

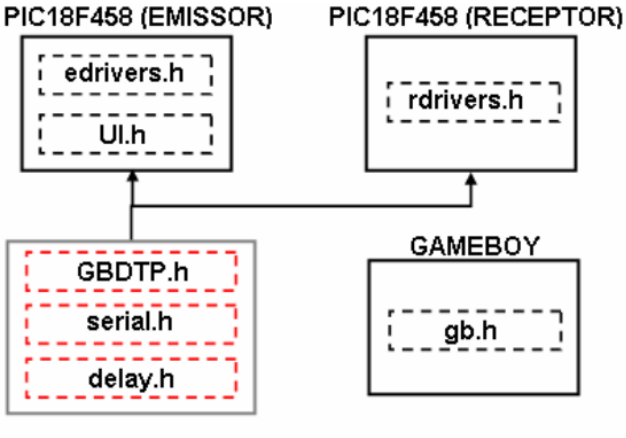


Figura 45: Bibliotecas Globais e Locais

Optou-se por fazer neste ponto a descrição das bibliotecas ditas globais, evitando assim a descrição das mesmas, sempre que utilizadas. Na tabela 8, está agrupada a informação relativa a cada biblioteca e respectivas funções.

BIBLIOTECA	FUNÇÃO	DESCRIÇÃO
GBDTP.h	`put_GBDTP()`	Envia pela USART o cabeçalho do GBDTP, que se encontra nesse instante na estrutura FRAME do tipo GBDTP.
	`get_GBDTP()`	Recebe via USART o cabeçalho do GBDTP, faz sincronismo de frame, e depois dos dados do cabeçalho se encontrarem validados, actualiza a estrutura FRAME do tipo GBDTP.
serial.h	`init_comms()`	Inicializa a USART, tendo em conta a Baud Rate defenida, e o respectivo modo de comunicação.
	`putch(char)`	Coloca um byte em TXREG da USART, para envio.
	`char getch()`	Faz polling em RCREG da USART, para recepção de um byte.
delay.h	`DelayUs(x)`	Espera *x* microsegundos.
	`DelayMs(n)`	Espera *n* milisegundos.

Tabela 8: Descrição das Bibliotecas globais implementadas

5.4. Software do Emissor

O software desenvolvido no emissor, foi escrito no modo compatível, ou seja apesar de ter sido orientado para o PIC18F4X8, pode também ser aplicado num PIC16F877, isto porque os níveis de prioridade não são utilizados e estão desactivados por software. A única atenção a ter em conta será na baud rate máxima, que no caso do PIC16F877, é de 57600bps, e no caso do PIC18F4X8, é 115200bps.

Como não existe no receptor memória suficiente para fazer buffering de dados, o algoritmo implementado, usa pacotes independentes para o áudio e para uma imagem. A principal característica deste algoritmo, consiste em enviar os dados que num dado instante de tempo que são considerados mais relevantes. Para isso, partiu-se do princípio, que na ausência de som, não faz sentido estar a enviar dados relativos ao áudio, o que evita uma sobrecarga da rede. Nesta circunstância a frame rate é máxima e constante. Quando a energia relativa a uma janela de som é superior a um dado limiar, então são enviados pacotes de som enquanto a energia estiver acima do limiar. Quando a energia da janela desce abaixo do limiar, é enviada uma imagem. Este algoritmo, tem uma vantagem, já que se adapta, em termos probabilísticos a um sinal de voz. Um sinal de voz é caracterizado por paragens entre as palavras. Se tivermos em conta que, o tempo entre duas palavras é uma eternidade quando comparado com o tempo de instrução do PIC, podemos inferir que é teoricamente possível ler uma imagem, e envia-la nos tempos de pausa do som.

O diagrama de fluxo seguinte, ilustra o algoritmo implementado no controlo do emissor.

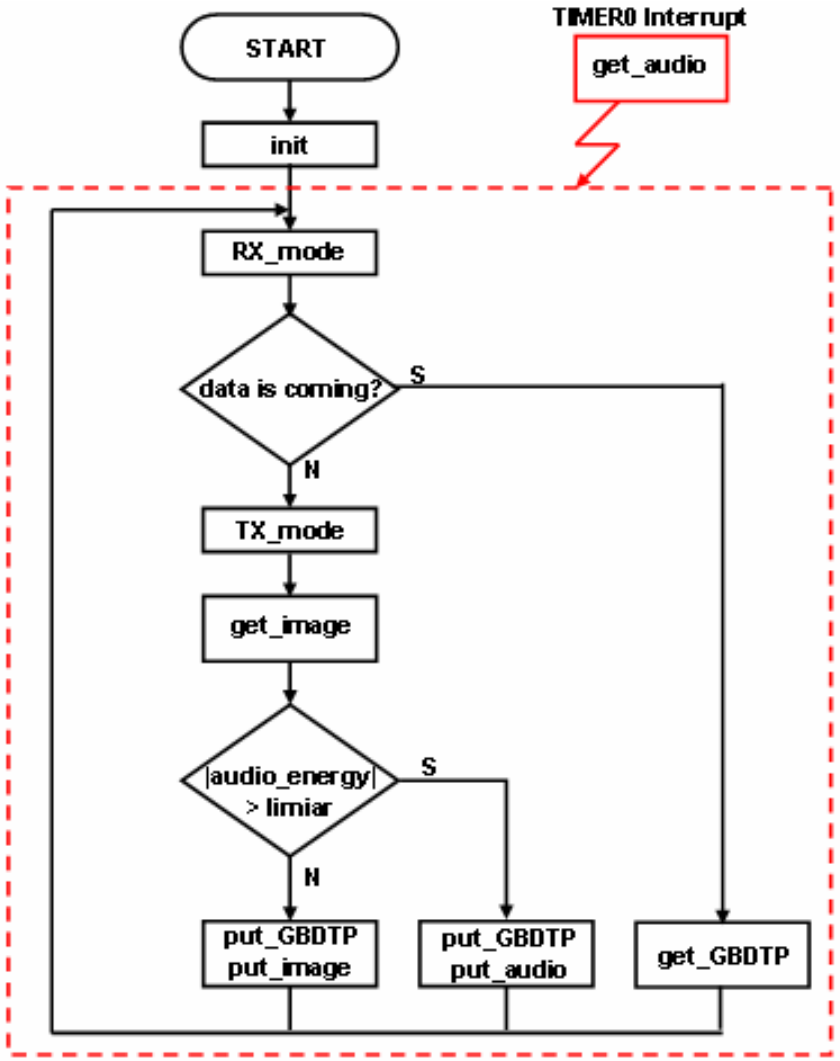


Figura 46: Diagrama de fluxo do software do emissor

Para descrever o funcionamento do algoritmo do software do emissor, a que o diagrama de fluxo diz respeito, usou-se uma notação na óptica do programador, de fácil entendimento para gerar o seguinte pseudo código:

```
10      init()          : inicia todos os Portos
20      RX_mode()       : altera o transceiver para o modo de recepção
                          escuta o canal de transmissão (Polling a CD\)
                          existem dados a chegar? Sim, então GOTO 90
30      TX_mode()       : altera o transceiver para o modo de emissão
40      get_image()     : lê uma imagem
                          a Energia do áudio > Limiar ? Sim, então GOTO 70
50      put_GBDTP(): coloca o header do GBDTProtocol para a imagem
60      put_image()     : envia os dados da imagem e GOTO 20
70      put_GBDTP(): coloca o header do GBDTProtocol para o áudio
80      put_audio()     : envia os dados do áudio e GOTO 20
90      get_GBDTP(): recebe um pacote de controlo e GOTO 20

RSI     get_audio()     : lê amostra de audio, RSI periódica (T=125us)
```

5.4.1. Bibliotecas Locais Implementadas no emissor

As bibliotecas locais, apresentadas em seguida foram desenvolvidas especificamente para o PIC do emissor. A biblioteca edrivers.h, apresenta todos os device drivers utilizados para controlar e aceder aos respectivos periféricos. A biblioteca UI.h, implementa um interface rudimentar, com o utilizador através da porta série, permitindo aceder a vários parâmetros que controlam o emissor.

BIBLIOTECA	FUNÇÃO	DESCRIÇÃO
edrivers.h	`init_ports()`	Inicializa todos os portos.
	`init_interrupts()`	Inicializa as interrupções.
	`poke(memByte, pos)`	Acesso à RAM Externa, coloca *memByte* na posição de memoria *pos*.
	`peek(pos)`	Acesso à RAM Externa, devolve o byte que se encontra na posição *pos*.
	`setReg(address, value)`	Programa o registo da AR, cujo endereço é *address*, e valor *value*.
	`bit_add(address , offset)`	Devolve o bit correspondente a *offset* de *address*, para a programação série da do endereço do registo da AR.
	`bit_val(value, offset)`	Devolve o bit correspondente a *offset* de *value*, para a programação série do valor do registo da AR.
	`in_pix()`	Devolve um pixel na resolução máxima, 8 bits.
	`inport_Byte(res)`	Devolve um Byte de pixeis, correspondente à resolução *res*.
		Detecta uma portadora no transceiver.

	C_Detect()	
UI.h	menu()	Devolve através da USART o menu principal de interface com o utilizador.
	menu_AR()	Devolve através da USART o menu com o actual setup dos registos da AR, permitindo alterações.
	menu_GBDTP()	Devolve através da USART o menu correspondente aos dados do protocolo que se encontram na estrutura de dados FRAME do tipo GBDTP.
	menu_res()	Devolve através da USART o menu correspondente à resolução actual utilizada, permitindo alterações.
	UI_menus()	Gere o acesso aos menus, em árvore, através do interface com o utilizador.

Tabela 9: Descrição das Bibliotecas locais do emissor

5.4.2. Interface com o utilizador do Emissor

Este interface, permite através da USART, alterar vários parâmetros do emissor. Utilizando um vulgar terminal de RS232, e utilizando a porta de série disponível no emissor, é possível aceder directamente às estruturas de dados que controlam a Artificial Retina, e à frame do GBDTP, ou alterar a resolução utilizada. Para aceder ao menu principal, basta enviar via RS232, para o emissor o carácter 'm'.

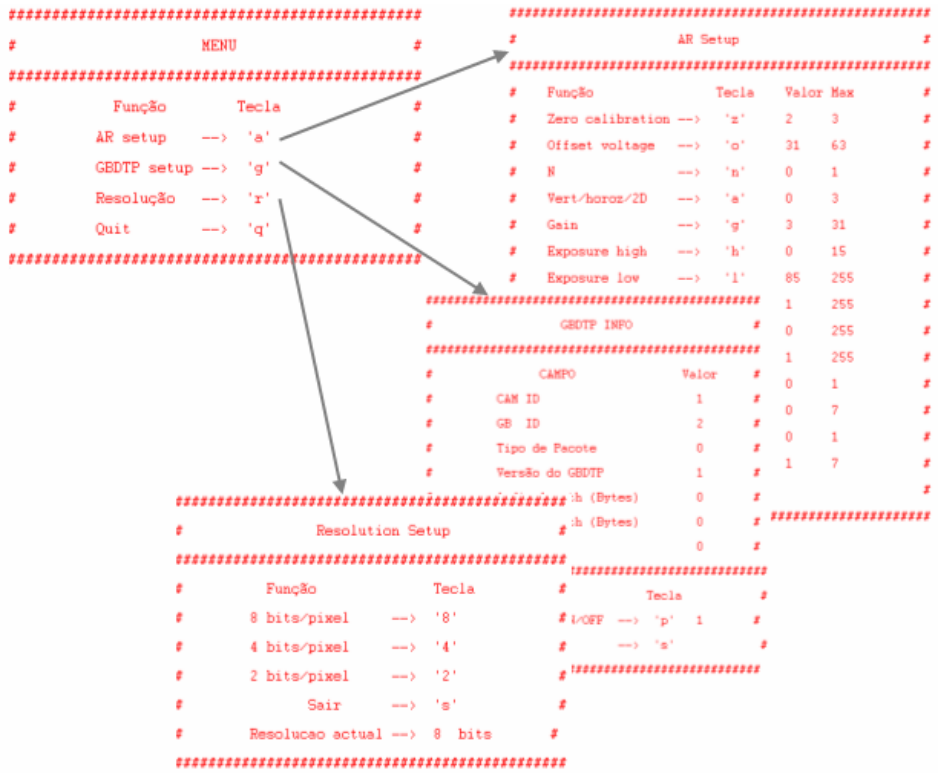


Figura 47: Menus de interface, via RS232, do emissor

5.5. Software do Receptor

Para implementar o receptor, foi necessário desenvolver um algoritmo de interface entre o PIC e o GB. A principal dificuldade residiu no facto de que, ambos os processadores são autónomos, e independentes. No arranque do sistema cada um começa a executar as suas tarefas, de um modo completamente assíncrono. A solução foi, a implementação de um sistema concorrente, em que diferentes tarefas, são lançadas por interrupção.

5.5.1. Software para o PIC18F458, do receptor

O software desenvolvido para o PIC do receptor, foi também escrito no modo compatível. O algoritmo implementado é baseado num sistema, concorrente, cujas tarefas são lançadas por interrupção, num sistema de prioridades fixas. Esta solução é imposta pelo tipo de comunicação utilizado entre o emissor-receptor. As comunicações estabelecidas através da USART são assíncronas, o que coloca problemas quanto à recepção dos dados, devido à sua imprevisibilidade. O diagrama de fluxo seguinte, ilustra o algoritmo implementado no controlo do emissor.

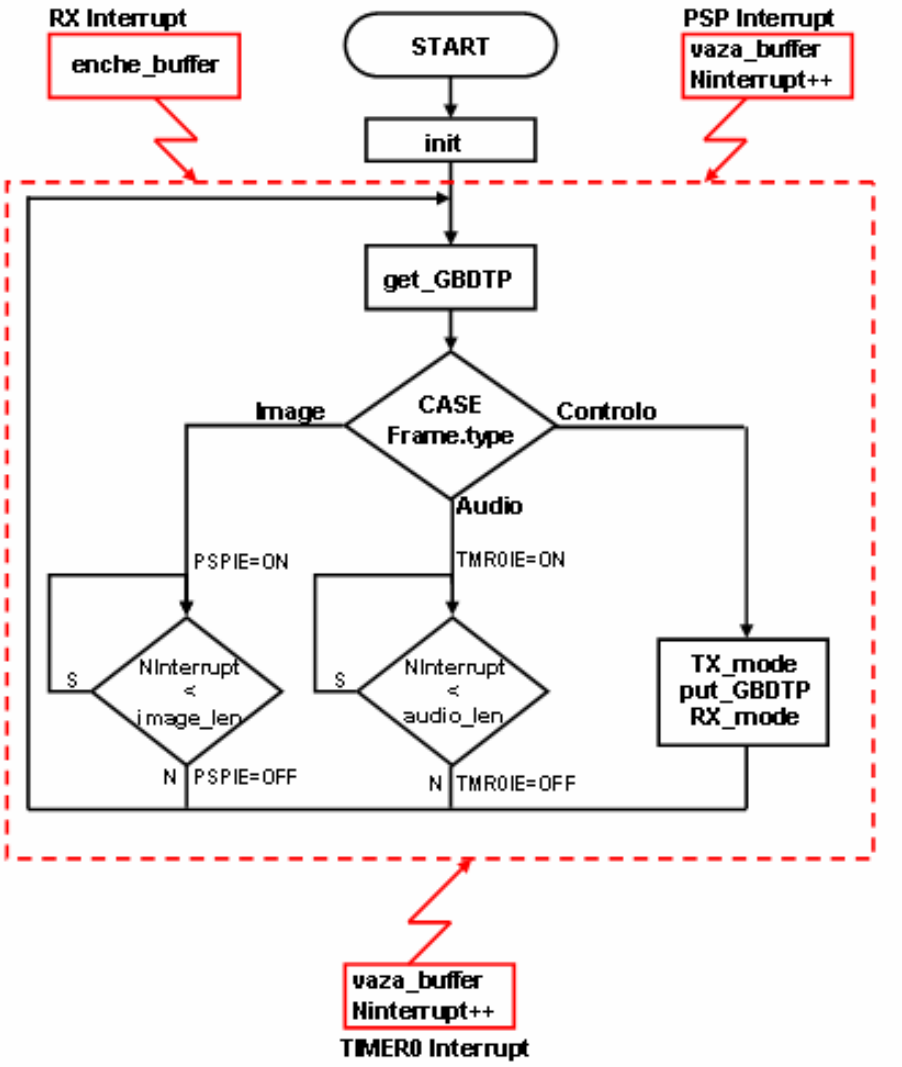


Figura 48: Diagrama de fluxo do software do PIC, no receptor

As tarefas concorrentes, podem ser ordenadas quanto à sua prioridade da seguinte maneira:

Prioridade	RSI	Função
máxima	RX	Chega de dados da USART
média	PSP	Leitura de dados de Imagem pelo GB
mínima	TIMER0	Alimenta a DAC a 8KHz

Tabela 10: Tabela de prioridades das tarefas no emissor.

Para descrever o funcionamento do algoritmo do software do receptor, a que o diagrama de fluxo diz respeito, aplicou-se a mesma notação usada anteriormente para gerar o seguinte pseudo código:

```
10      init          : inicia todos os Portos
20      get_GBDTP     : extrai o header do GBDTProtocol
                        no caso de se tratar de dados da IMAGEM, GOTO 30
                        no caso de se tratar de dados do AUDIO, GOTO  50
                        no caso de se tratar de dados de CONTROLO, GOTO 70
30      PSPIE ON      : Liga as interrupções da PSP
                        : Nº de Interrupções < Image length ? Não, então GOTO 30
40      PSPIE OFF     : Desliga as interrupções da PSP e GOTO 20
50      TMR0IE ON     : Liga as interrupções do TIMER0
                        : Nº de Interrupções < Audio length ? Não, então GOTO 50
60      TMR0IE OFF    : Desliga as interrupções do TIMER0 e GOTO 20
70      TX_mode       : altera o transceiver para o modo de emissão
80      put_GBDTP     : coloca o header do GBDTProtocol para a áudio
                          envia o respectivo comando
90      RX_mode       : altera o transceiver para o modo de recepção e GOTO 20

RSI1    enche_buffer  : enche o buffer circular, com os dados que chegam via
                          RS232.
RSI2    vaza_buffer   : o GB, lê um byte do buffer circular
                        Incrementa o contador de interrupções.
RSI3    vaza_buffer   : é colocado um byte do buffer circular na DAC
                        Incrementa o contador de interrupções.
```

5.5.1.1. Bibliotecas Locais Implementadas no receptor

As bibliotecas locais, apresentadas em seguida foram desenvolvidas especificamente para o PIC do receptor. A biblioteca rdrivers.h, apresenta todos os device drivers utilizados para controlar e aceder aos respectivos periféricos.

BIBLIOTECA	FUNÇÃO	DESCRIÇÃO

rdrivers.h	`init_ports()`
	
	`init_TIMER0()`
		`Inicializa todos os portos.`
		
		`Inicializa o TIMER0 para alimentar a DAC.`

Tabela 11: Descrição das Bibliotecas locais do receptor (PIC) implementadas

5.5.1.2. Dimensionamento do Buffer Circular

Como estamos perante um sistema concorrente, em que as tarefas são lançadas por interrupção, é indispensável fazer buffering dos dados já que as interrupções ocorrem a ritmos diferentes. A solução encontrada consistiu em implementar um buffer circular, que permita, a uma tarefa escrever no buffer, enquanto outra lê, em posições diferentes. O dimensionamento do buffer, consistiu em estimar o pior caso, correspondente à leitura de uma imagem completa. Em seguida temos uma tabela que apresenta o período de interrupção para as duas taxas de transmissão mais importantes.

Baud Rate	TX Real= B. R.*0,8	T de Interrupt	Enche o buffer (256Bytes)
57600bps	5760Bytes/s	~173µs	44,3ms
115200bps	11520Bytes/s	~87µs	22,3ms

Tabela 12: Período de Interrupção da USART

A solução implementada parte do período de interrupção da USART, para taxas de transmissão pré-defenidas. Sincronizou-se o GB, para uma frequência de leitura aproximadamente igual à da chegada de dados pela porta série, com um jitter máximo de 5µs, que corresponde no pior caso, a um tempo de 3584*5µs = 17,9ms como este valor é menor que o tempo que o buffer demora a encher para uma Baud Rate igual a 115200bps (22,3ms), pode garantir-se que o tamanho do buffer é suficiente para garantir que o GB leia uma imagem sem problemas. A figura seguinte representa a solução encontrada.

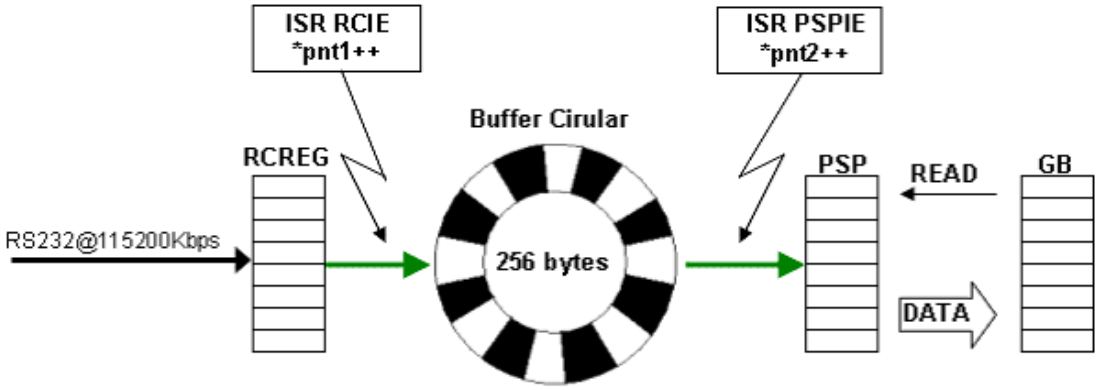


Figura 49: Dimensionamento do Buffer Circular

5.5.2. Software para o GameBoy

O software desenvolvido para o GameBoy™, foi escrito em C e compilado com o GBDK (GameBoy Development Kit) para DOS. Como este compilador é relativamente rudimentar, optou-se por escrever o código, e respectivas funções no mesmo ficheiro. O algoritmo implementado no GameBoy™, está direccionado para a aplicação, sendo privilegiada a relação com o utilizador, através de menus. A aquisição de uma imagem por parte do GB, está sinalizada no diagrama de fluxo pelo picotado a vermelho.

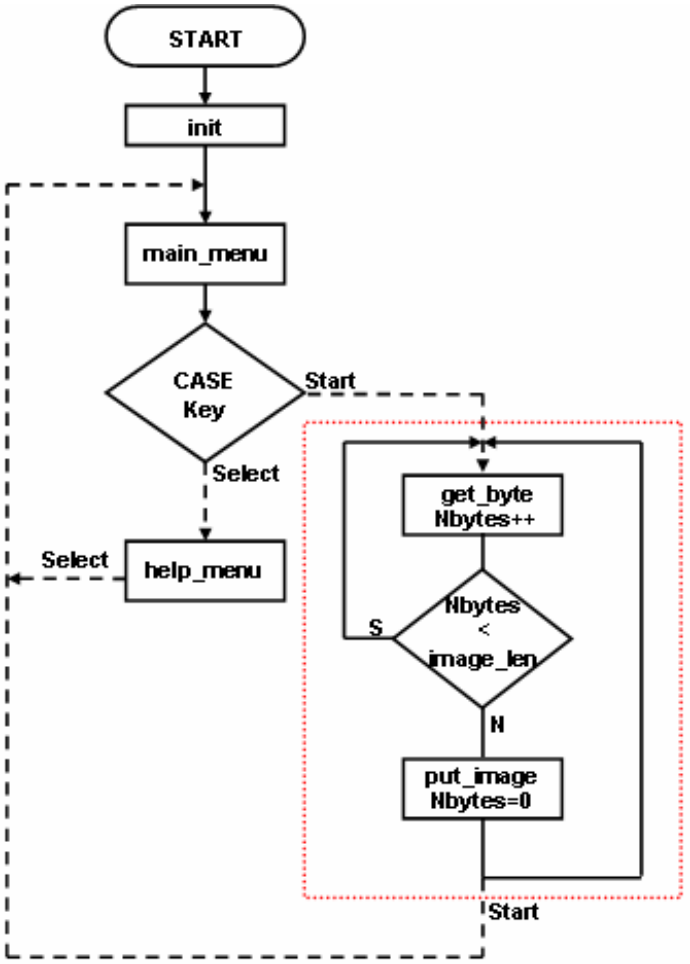


Figura 50: Diagrama de fluxo do software do GB, no receptor

Para descrever o funcionamento do algoritmo do software implementado no GB, aplicou-se mais uma vez a notação usada anteriormente para gerar o seguinte pseudo código:

```
10      init          : arranque com o logo da UA
20      main_menu     : menu principal
                        no caso do utilizador pressionar em START, GOTO 30
                        no caso do utilizador pressionar em SELECT, GOTO
30      get_byte      : lê um byte do PIC
                        incrementa o contador de bytes, Nbytes
                        Nbytes < image length , então GOTO 30
40      put_image     : coloca a imagem na VRAM
                        START foi pressionado? Então GOTO 20
                        inicializa o contador de bytes e GOTO 30
```

As funções implementadas para desenvolver o software do GameBoy™, são apresentadas na seguinte tabela:

Função	Descrição
load_pal()	Carrega para a VRAM, as paletes de cor usadas, os 16 tons de cinza e as paletes para o interface gráfico.
UA_logo()	Envia para a VRAM, o logo da Universidade de Aveiro
start_fonts()	Inicializa o sistema de fontes
menu()	Carrega para a VRAM o menu principal
help()	Carrega para a VRAM o menu de help
Inport()	Lê um byte do PIC, neste caso da zona que se fez a descodificação de endereços, de A000 até BFFF
outport(data)	Escreve o byte *data*, no PIC, neste caso da zona que se fez a descodificação de endereços, A000 até BFFF
put_image()	Coloca a imagem que foi lida do PIC, byte a byte, na VRAM

Tabela 13: Funções implementadas para o software do GameBoy™

5.5.2.1. Interface com o utilizador

O interface com o utilizador é nesta fase muito simples, dados os objectivos propostos. As teclas de Start e Select do GameBoy™, correspondem a início da comunicação e menu de ajuda, respectivamente. Estas imagens foram captadas com recurso ao simulador No$gmb, já que se tornou impossível fotografar o display do GB, este é brilhante e funciona com um espelho.

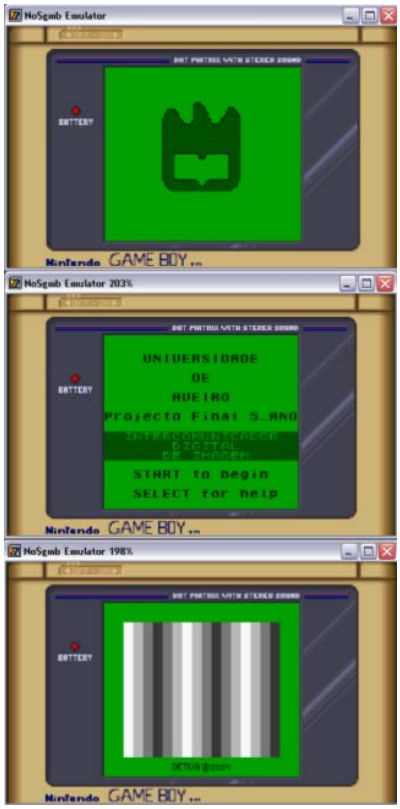


Figura 51: Interface com o utilizador do GameBoy™

5.6. Software do Interface para o PC

Como a ideia do projecto é criar uma pequena LAN doméstica considerou-se uma mais valia desenvolver uma aplicação em Visual Basic para a recepção da imagem num ou em vários PC's. Esta aplicação visa uma utilização bastante simples por parte do utilizador. O PC, nesta fase, assume apenas o papel dum receptor passivo, isto é, apenas recebe os pacotes enviados da câmara não podendo enviar qualquer comando de configuração para alterar os parâmetros de dos registos do emissor.

Em seguida é apresentado o diagrama de fluxo do software da aplicação que serve de interface entre o módulo do emissor e o PC.

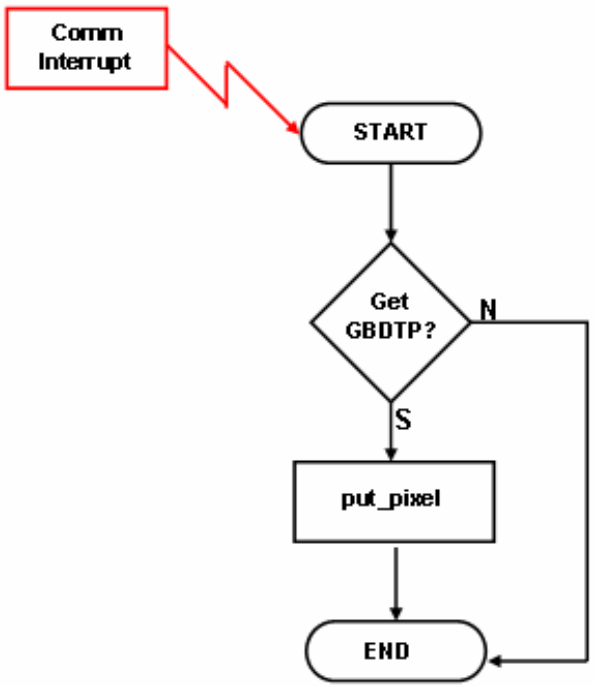


Figura 52: Diagrama de fluxo da aplicação para o PC

No software desenvolvido, quando a USART recebe um byte é gerada uma interrupção que origina o processamento desse byte. No caso de ainda não se ter recebido nenhum valor válido na porta série, o programa fica à espera de receber o dos bytes de sincronismo do GBDTP. Depois de interpretados com sucesso os bytes de sincronismo, o programa passa a actualizar os dados do GBDTP, na estrutura interna. Se o destino do pacote for o PC em questão, os dados são então colocados no ecrã e a imagem começa a surgir. Quando a imagem terminar, o programa fica à espera de receber um novo pacote, repetindo-se todo o procedimento que foi descrito.

5.6.1. Rotinas implementadas

Passa-se a apresentar as várias rotinas implementadas para a recepção de dados da porta série, seu processamento e apresentação da imagem no ecrã.

- **Sub-Rotina de arranque da aplicação (Private Sub Form_Load()):**

  Esta rotina de arranque do programa, tem como objectivo inicializar as variáveis necessárias para o arranque dos eventos. Nela é carregada toda a informação do ecrã de apresentação ao utilizador, bem como os valores iniciais das variáveis do programa, tais como activar os valores desejados para a porta série, inicializar a Baud Rate e definir que se pretende iniciar com a recepção de imagem com zoom de 4 vezes o tamanho da imagem captada.

- **Sub-Rotina para iniciar e terminar a recepção dos dados pela porta série (Private Sub botao_click()):**

  Esta rotina serve para, no momento em que o utilizador carrega no botão de "POWER ON/ POWER OFF", detectar se o computador tem a porte série "aberta" (terminologia usada no Visual Basic), isto é, se está a receber dados, ou se está "fechada", isto é não recebe dados por imposição do software. No caso de ter a porta série activa esta rotina é encarregue de a desactivar e vice versa.

- **Sub-Rotina encarregue de processar os dados recebidos da porta série (Private Sub MSComm1_OnComm()):**

  Rotina encarregue da validação de recepção e do processamento dos dados da porta série.
  Esta rotina é a responsável por todo o processamento dos bytes recebidos pela porta série, identifica se são bytes validos, interpreta-os segundo o GBDTProtocol e decide o que fazer com o byte recebido.
  No caso do Byte pertencer a um comando de controlo, guarda-o na variável adequada, no caso de ser um byte de imagem, chama outra sub-rotina para o enviar para o ecrã.
  No caso de ser um byte não valido, para o protocolo, ele é descartado e a sub-rotina é activada por interrupção aquando a recepção dum novo byte.

- **Sub-Rotina de impressão da imagem no ecrã com zoom (Private Sub ecran(pixel As Integer)):**

  Esta rotina faz um zoom de quatro vezes da imagem que recebe, isto é, cada byte que recebe referente à imagem, processa-o como se tivéssemos recebido 4 bytes e imprime-os no ecrã.

- **Sub-Rotina de impressão da imagem sem zoom no ecrã (Private Sub ecransemzoom(pixel As Integer)):**

  Esta rotina está encarregue de imprimir no ecrã a imagem recebida do emissor. Imprime a imagem no tamanho com que e recebido pela "Artificial Retina" ou seja 128 x 112 pixeis.

- **Sub-Rotina de configuração da COM 1 (Private Sub COM1_Click()):**

Rotina de activação da COM 1, esta rotina prepara a aplicação para receber os dados da porta série pela COM 1.

- **Sub-Rotina de configuração da COM 2 (Private Sub COM2_Click()):**

  Rotina de activação da COM 2, esta rotina prepara a aplicação para receber os dados da porta série pela COM 2.

- **Sub-Rotina de configuração da Baud-Rate (Private Sub baudrate_click()):**

  Rotina que configura o formato e a velocidade de recepção dos dados recebidos pela porta série. Na aplicação há a possibilidade de escolher a Baud Rate. Esta rotina tem a função de actualizar os parâmetros de recepção para essa nova Baud Rate.

- **Sub-Rotina de abandono da aplicação (Private Sub Form_Unload(Cancel As Integer)):**

  Esta rotina tem a função de prevenir que a porta série não fique activa no caso dum "abandono secundário" da aplicação, ou seja cada vez que se fechar a janela da aplicação sem ser através do botão correspondente (botão "QUIT") não se corre o risco de se ficar com a porta série activa.

- **Sub-Rotina de saída da aplicação (Private Sub sair_Click()):**

  Rotina que tem como função desactivar a porta série e abandonar a aplicação quando o utilizador "clica" no botão "QUIT".

5.6.2. Interface com o utilizador

Apresentação e descrição dos vários campos do layout da interface da aplicação com o utilizador.

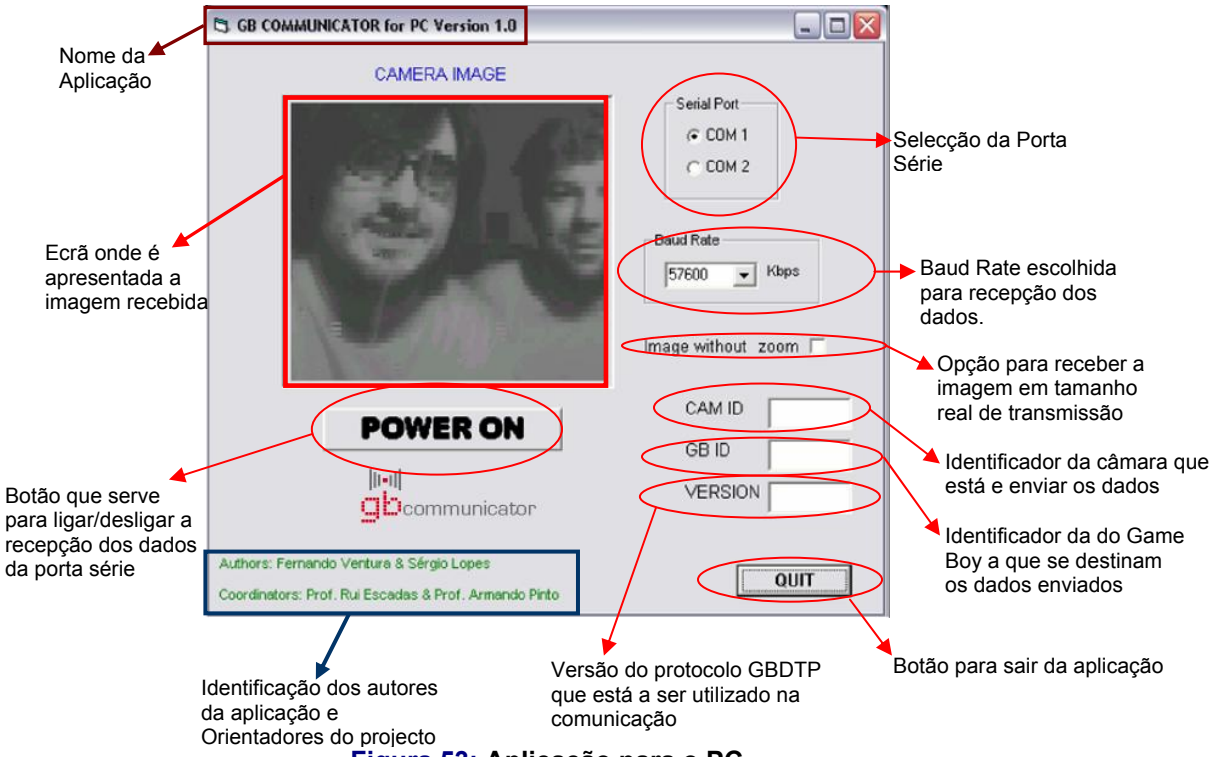


Figura 53: Aplicação para o PC

Na interface da aplicação, de recepção de dados do emissor pelo PC, com o utilizador tem-se uma janela como a que é apresentada na figura anterior, onde se pode encontrar o cabeçalho, GB COMMUNICATOR for PC version 1 (nome da aplicação). Nesta interface com o utilizador encontra-se um campo onde se pode escolher a porta série à qual está ligado o receptor. As portas podem ser, ou a COM 1 ou a COM 2. Pode também escolher-se qual a Baud Rate a que se pretende receber os dados, ela pode ser de 9600, 19200, 38400, 57600 ou 115200 Kbps, tem de ter-se em atenção que esta aplicação funciona apenas em modo passivo, ou seja não comunica com o emissor, por isso a Baud Rate escolhida tem de ser igual à do emissor.

Outra das possibilidades é a da escolha da recepção da imagem em tamanho real, isto é, receber a imagem tal como ela é enviada pela câmara (tamanho 128 pixeis x 128 pixeis) para tal basta seleccionar a opção “Image without zoom”. Por defeito a aplicação mostra ao utilizador uma imagem ampliada quatro vezes.

Em seguida podem encontrar-se três campos, um que identifica a câmara que está a enviar os dados (CAM ID), outro para que consola a câmara está a enviar (GB ID) e um terceiro que identifica qual a versão do protocolo que a câmara está a utilizar. Estes campos são interessantes uma vez que o intercomunicador implementado tem a potencialidade de funcionar com várias câmaras e vários receptores (Game Boys) e assim fica a saber-se quem são os intervenientes da comunicação que se está a efectuar.

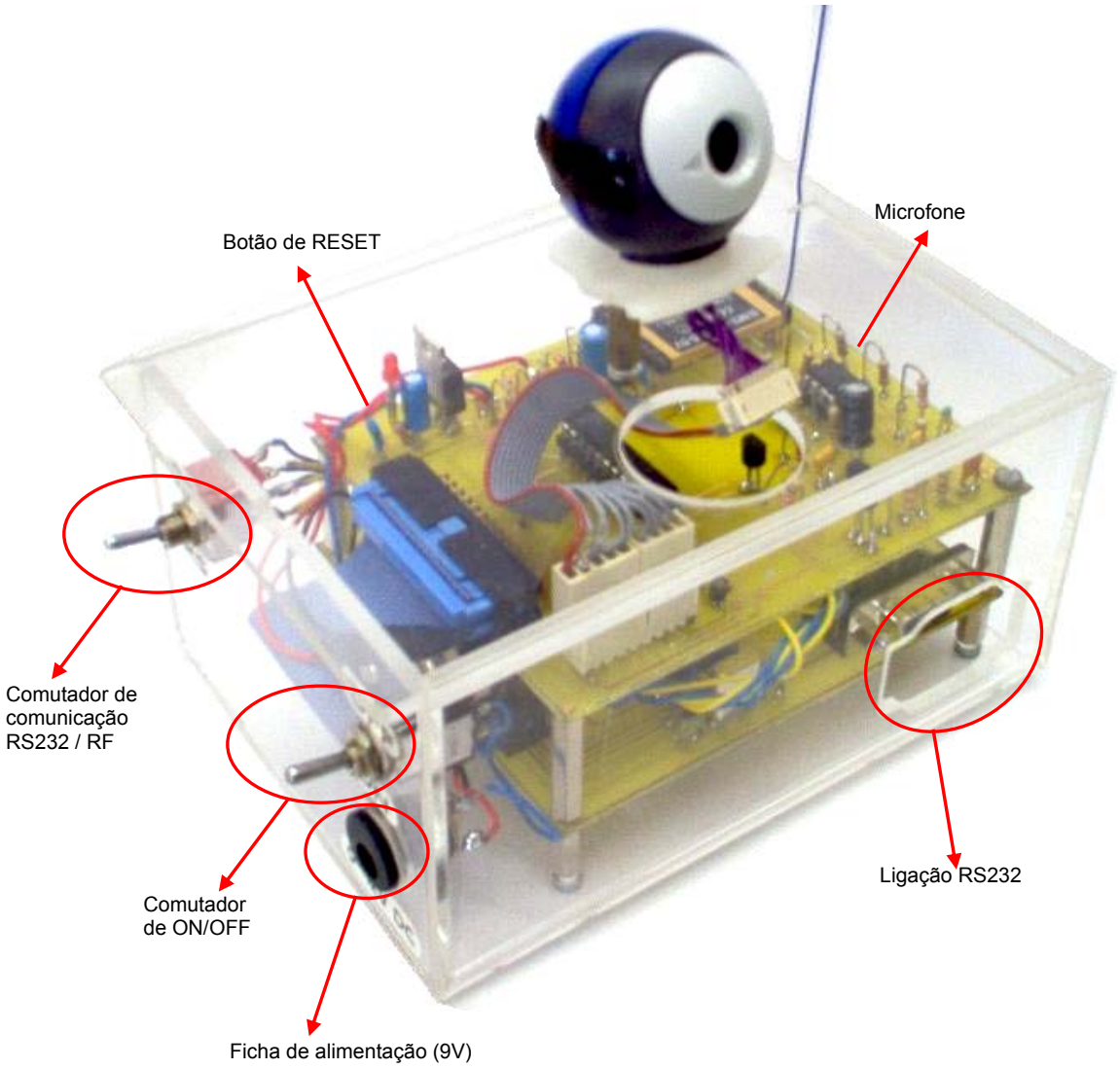
A aplicação tem o botão “QUIT” que serve para o utilizador sair da aplicação e o botão “POWER ON/POWER OFF” que serve para o utilizador começar/terminar a recepção dos dados do emissor. Se no botão se ler “POWER ON” significa que a aplicação não está a receber dados e que passa a recebê-los no caso de se “clicar” em cima dele, caso contrário, ou seja, no caso em que se lê “POWER OFF” significa que se estão a receber dados do emissor. No caso de se querer deixar de receber dados, basta “clicar” em cima do botão e volta a aparecer “POWER ON”.

Como campo principal da aplicação, tem-se a “moldura” onde se recebem as imagens enviadas pelo emissor. Finalmente, no canto inferior esquerdo temos a identificação dos autores da aplicação/projecto bem como o nome dos orientadores do projecto.

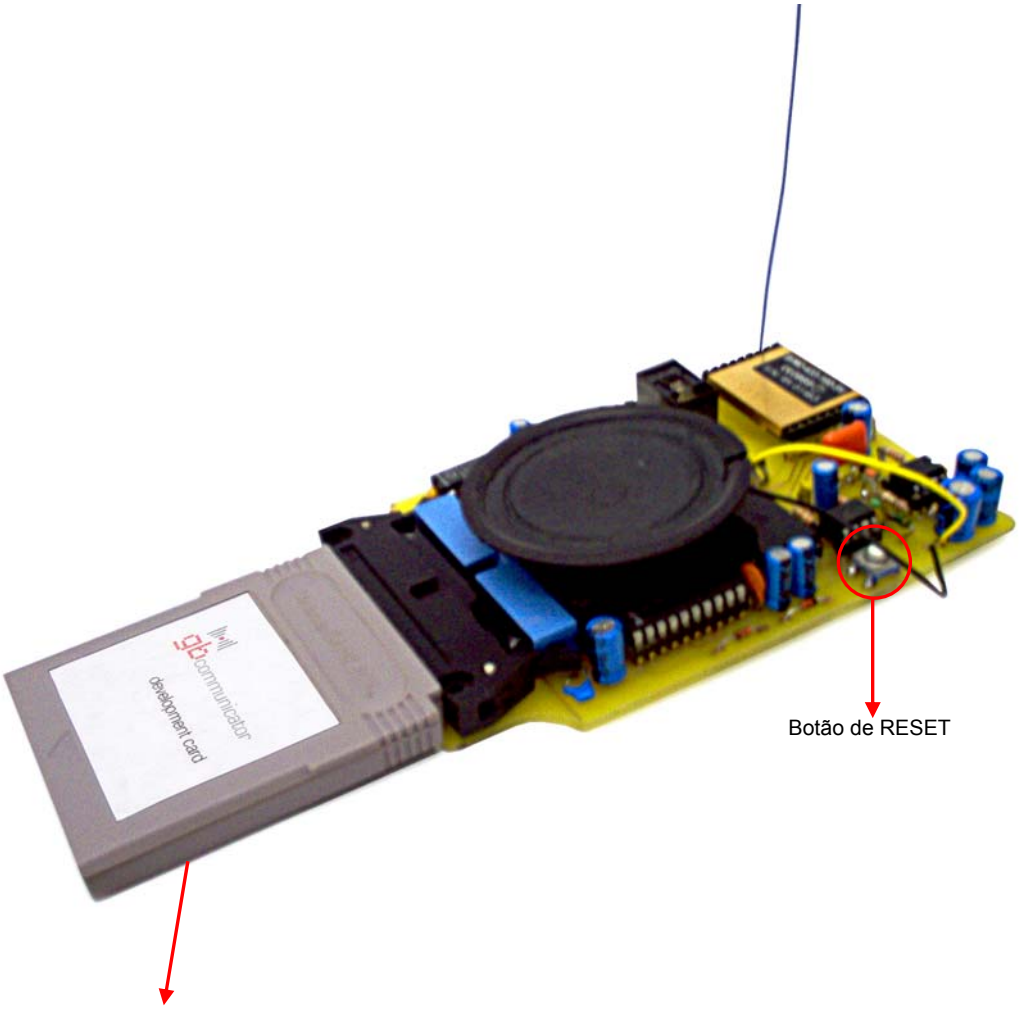
6. Protótipo

Neste ponto pretende-se apresentar o protótipo implementado, com o intuito de descrever as suas funcionalidades e relatar o respectivo modo de funcionamento. Pode-se considerar, a seguinte descrição como um pequeno manual do utilizador, dos vários elementos construídos.

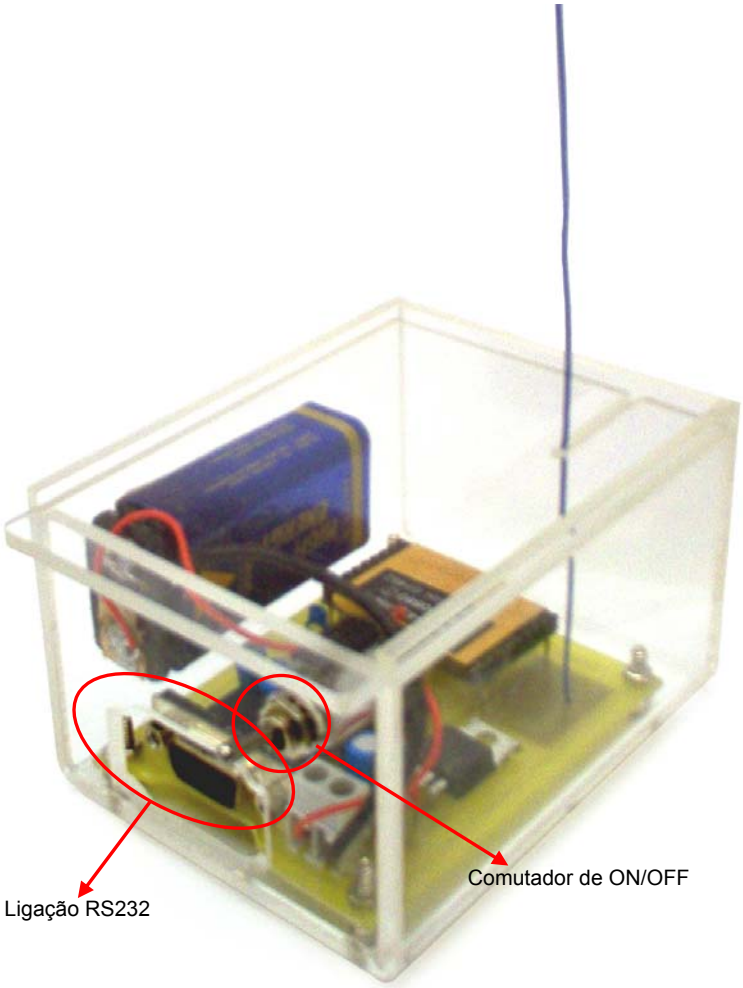
6.1. Emissor



6.2. Receptor


development card
Botão de RESET
Cartucho de
desenvovimento

6.3. Interface para o PC



6.4. Orçamento do protótipo

Um projecto, tem na fase de desenvolvimento custos acrescidos, isto porque, nessa fase, são normalmente tentadas várias soluções. Este orçamento, tem como principal objectivo, listar o material usado e controlar os custos de implementação do protótipo. Optou-se por dividir os custos pelos três elementos implementados, o emissor, o interface para o PC e o receptor, dando origem aos três orçamentos apresentados a seguir:

Orçamento do emissor:

Designação	Quantidade	Preço Unitário	Preço
Câmara CCD M64282FP	1	16,81	16,81
Transceiver BiM2-433-160-5V	1	23,36	23,36
Comutador ON/OFF	2	0,92	1,84
Ficha de alimentação	1	0,76	0,76
Ficha 26 pinos (macho)	2	0,40	0,80
Ficha DB9	1	0,25	0,25
LED	1	0,09	0,09
Ficha p/ pilha de 9 V (snap)	1	0,19	0,19
Resistência 1/4 W	17	0,03	0,51
Flat cable c/ 2 fichas 26 pinos (fêmea)	1	0,70	0,70
Condensador	22	0,05	1,10
Cristal de 10 Mhz	1	0,50	0,50
Transístor BC557C	2	0,06	0,12
Regulador de tensão TL431C	1	0,75	0,75
ADC 0820CNEN	1	1,68	1,68
Díodo	2	0,12	0,24
Botão de Reset	1	0,25	0,25
PIC Microchip 18F458	1	6,58	6,58
CMOS SRAM 8k*8bit HY6264A	1	3,46	3,46
CMOS OpAmp MCP602	1	0,51	0,51
Regulador de Tensão MC7805C	1	0,30	0,30
SN74HC373N	1	0,30	0,30
Ficha 9 pinos p/ câmara (macho)	1	0,19	0,19
Ficha 9 pinos p/ câmara (fêmea)	1	0,17	0,17
MAX 232	1	1,62	1,62
Placa de Circuito Impresso	2	8,40	16,80
Caixa Acrílico 13,5x7,3x10cm	1	10,50	10,50
Socket p/ os Circuitos Integrados	7	0,08	0,56
TOTAL			**90,94**
IVA			17,28
Preço c/ IVA			**108,22**

Tabela 14: **Orçamento do emissor**

Orçamento do interface para o PC:

Designação	Quantidade	Preço Unitário	Preço
Transceiver BiM2-433-160-5V	1	23,36	23,36
Comutador ON/OFF	1	0,92	0,92
Ficha DB9	1	0,25	0,25
Regulador de Tensão MC7805C	1	0,30	0,30
MAX 232	1	1,62	1,62
Resistência 1/4 W	1	0,03	0,03
Conector p/ pilha de 9 V (snap)	1	0,19	0,19
Condensador	8	0,05	0,40
LED	1	0,09	0,09
Placa de Circuito Impresso	1	8,40	8,40
Caixa Acrílico 9x5,5x7,2cm	1	10,50	10,50

TOTAL	**46,06**
IVA	8,75
Preço c/ IVA	**54,82**

Tabela 15: Orçamento do interface para o PC

Orçamento do receptor:

Designação	Quantidade	Preço Unitário	Preço
Consola Game Boy SP	1	139,00	139,00
Transceiver BiM2-433-160-5V	1	23,36	23,36
PIC Microchip 18F458	1	6,58	6,58
Condensador	27	0,05	1,35
Altifalante	1	2,37	2,37
74HC10N	1	0,42	0,42
CMOS OpAmp MCP602	1	0,51	0,51
MAX232	1	1,62	1,62
Cristal 10MHz	1	0,50	0,50
Resistencia 1/4 W	12	0,03	0,36
Amplificador Áudio TBA820N	1	0,42	0,42
DAC 0832	1	1,65	1,65
Diodo 1N4148	4	0,12	0,48
Placa de Circuito Impresso	1	8,40	8,40
Socket p/ os Circuitos Integrados	7	0,08	0,56
TOTAL			**187,58**
IVA			35,64
Preço c/ IVA			**223,22**

Tabela 16: Orçamento do receptor

Os preços apresentados dos componentes, são para ser considerados como valores médios e estão o mais aproximado do valor comercial possível, contudo uma vez que a maior parte deste material foi adquirido no armazém do DET, não podemos apresentar o custo real dos componentes.

O valor total dos 3 módulos implementados é 386,26 (trezentos e oitenta e seis euros e vinte seis cêntimos). Este valor parece muito elevado comparando com outros produtos de idêntica utilização, mas temos de considerar que estamos a falar de preços para um protótipo. Todos os valores foram consultados como se tratasse duma compra unitária e como é sabido o preço diminui consoante a quantidade do material que se adquire. Quer-se com isto dizer que estes preços diminuem significativamente com construção em série do modelo.

Também se deve ter em consideração que os intercomunicadores que se encontram à venda nos supermercados, são sistemas fechados e apenas servem para recepção/envio de imagens e som, não havendo possibilidades para upgrades. No caso do sistema que se implementou, o utilizador tem a potencialidade de funcionar com o intercomunicador numa pequena LAN podendo utilizar o PC como receptor. Para além disso o GameBoy usado na implementação do receptor, mantém as suas propriedades de consola de jogos, para isso basta substituir o cartucho/Receptor, por um cartucho de jogos.

7. Calendarização das tarefas do projecto

De seguida apresentam-se as várias fases de desenvolvimento e implementação dos módulos e do software do intercomunicador. Esta apresentação, tem como objectivo identificar de grosso modo a sequência dos trabalhos para a elaboração do sistema.

Fases do Projecto	Set.	Out.	Nov.	Dez.	Jan.	Fev.	Mar.	Abr.	Mai.	Jun.	Jul.	
Estudo da Arquitectura do GameBoy												OK!
Dimensionamento do projecto												OK!
Estudo da Arquitectura e Protocolo, DECT												OK!
Estudo da Arquitectura PIC 18F4x8												OK!
Estudo do Transceiver BiM2-433-160												OK!
Implementação do Cartucho de desenvolvimento												OK!
Implementação do hardware do Emissor												OK!
Implementação do hardware do Receptor												OK!
Desenvolvimento do Software do Receptor (GB)												OK!
Desenvolvimento do Software do Emissor (PIC)												OK!
Desenvolvimento do Software do Receptor (PIC)												OK!
Desenvolvimento do Software do PC												OK!
Desenvolvimento do PCB do Emissor												OK!
Desenvolvimento do PCB do Receptor												OK!
Desenvolvimento do PCB do PC adapter												OK!
Testes de Integração												
Optimização do Software												OK!

Tabela 17: Calendarização das tarefas do projecto

Em seguida, são consideradas, e descritas as fases consideradas mais relevantes, no desenvolvimento do projecto. Um dos pontos de partida do projecto, é a utilização do GameBoy™ por isso, numa primeira fase pesquisaram-se o tipo de ferramentas que existem para o desenvolvimento do projecto. Procurou-se todo o tipo de informação acerca do hardware que constitui o GameBoy™, para se conhecerem as potencialidades e restrições desta consola. Pesquisou-se também, qual o software de desenvolvimento que se pode usar, tais como simuladores, compiladores e programadores de cartuchos para se poder programar a consola. Uma vez que já se tinha conhecimento da arquitectura interna do GameBoy™, iniciou-se a projecção e dimensionamento do sistema. A construção do cartucho de desenvolvimento consistiu no aproveitando de um cartucho da Nintendo que continha uma ROM com um jogo. Foi-lhe substituída essa ROM por uma flash ROM AMD29F040, uma vez que iria ser necessário carregar o GameBoy™ com o software da aplicação que se iria desenvolver.

Outra importante fase a considerar foi o estudo da arquitectura e protocolo DECT (Digital Cordless Telephone Technology). Este protocolo é utilizado nos vulgares telefones portáteis e as suas características são perfeitas para ser utilizado no âmbito deste projecto. O abandono deste sistema de comunicações, foi o resultado do elevado preço deste módulo de comunicações, indo de encontro directo a um dos principais objectivos deste projecto, construir um intercomunicador de baixo custo.

8. Conclusões e Recomendações

Tendo em conta que o projecto desenvolvido é uma disciplina anual, faz sentido relacionar os resultados obtidos com os objectivos iniciais propostos. No geral, pode afirmar-se que, todos os objectivos foram alcançados com relativo sucesso. Passa-se a descriminar com brevidade os mais importantes:

→ O principal objectivo do intercomunicador era criar um setup do tipo emissor-receptor, ideia essa que foi alargada, acabando por se desenvolver um sistema que permite várias comunicações ponto-a-ponto e comunicações ponto-multi-ponto.

→ Construiu-se o protótipo do Emissor do intercomunicador. Este cumpre as especificações inicialmente propostas, com algum ruído sistemático na aquisição da imagem. Este ruído deverá ter origem na ADC utilizada. O protocolo GBDTP, é aplicado no emissor para a comunicação com os nós do sistema, PC e receptor. A taxa de transmissão máxima, em RF, conseguida sem introdução de erros , foi de 57600bps, já que para taxas superiores, existe perda de sincronismo em algumas frames, corrompendo assim os dados.

→ Construiu-se o hardware para um interface com o PC. O emissor comunica, via RF, com o PC sem problemas, na resolução máxima (8bits/pixel) e com uma taxa de transmissão de 57600bps. Foi desenvolvida uma aplicação para o PC, que interpreta o protocolo GBDTP.

→ Para desenvolver o receptor, foi necessário construir um cartucho de desenvolvimento, que permite ter acesso ao barramento da consola, e acesso a memória de massa, para armazenar o código.

→ Construiu-se o hardware para o receptor, que faz o interface do emissor com o Gameboy™, a taxa de transmissão máxima conseguida entre o PC e o receptor foi de 57600bps. O protocolo funciona para pacotes alternados de áudio e de imagem com garantias de fiabilidade. As imagens enviadas do PC (2bit/pixel) para o Gameboy™ foram previamente filtradas com um filtro de dithering 2D.

→ Para resoluções mais baixas (em relação à profundidade de cor) um pré-processamento com um filtro de dithering 2D, melhora considerávelmente a qualidade da imagem.

Para finalizar, pedimos a palavra para dizer que estamos orgulhosos do trabalho desenvolvido durante este ano lectivo na disciplina mais exigente do nosso curso.

9. Bibliografia

- Mitsubishi CMOS Image Sensors & Digital Imaging Solutions, 1998
- JERKE, NOEL (1999) – The complete Reference Visual Basic 6, Berkeley, Osborne/McGraw-Hill.
- SCHILDT, HERBERT(1991) – C completo e total, São Paulo, Osborne/McGraw-Hill.
- KERNIGHAN, BRIAN W. & RITCHIE, DENNIS M. (1988) – The C Programming Language Second Edition, New Jersey, Prentice Hall.
- SEDRA, ADEL S. & SMITH, KENNETH C. (1998) – Microelectronic Circuits Fourth Edition, New York, Oxford University Press.
- DATASHEETS
  - TBA820 1.2W AUDIO AMPLIFIER
  - ADC0820 8-Bit High Speed µP Compatible A/D Converter
  - Mitsubishi Integrated Circuit M64282FP Image Sensor (Artificial Retina LSI)
  - DAC0830/DAC0832 8-Bit µP Compatible, Double-Buffered D to A Converters
  - 74HC/HCT373 Octal D-type transparent latch with 3-state
  - MAX232, DUAL EIA-232 DRIVERS/RECEIVERS
  - 74HC/HCT10 Triple 3-input NAND gate
  - MCP601/2/3/4; 2.7V to 5.5V Single-Supply CMOS Op Amps
  - PIC18FXX8; 28/40-Pin High Performance, Enhanced FLASH Microcontrollers
  - MC7800 Series VOLTAGE REGULATOR
  - TL431 ADJUSTABLE PRECISION SHUNT REGULATORS
  - Am29F040B; 4 Megabit (512 K x 8-Bit) CMOS Flash Memory
  - HY6264A Series; 8Kx8bit CMOS SRAM
  - Radiometrix BiM2-433-160 transceiver

10. Sites relevantes

- Sites de componentes de electrónica
  http://forum.microchip.com/
  http://www.microchip.com/
  http://www.devrs.com/pic
  http://www.analog.com/
  http://detua-doc.det.ua.pt/
  http://www.microchip.com/
  http://www.national.com/
  http://www.ti.com/
  http://www.amd.com/us-en/
  http://www.leiritronica.com
  http://www.cec-coimbra.pt
  http://www.semiconductors.philips.com/
  http://www.mikroelektronika.co.yu/

- Sites relacionados com as comunicações

  http://www.radiometrix.eu.com/html/products/bim2_6.html
  http://www.lextronic.fr/Hybrides/Radiometrix/bim2.htm
  http://www.hoeft-wessel.com/en/products/HW86910.htm
  http://www.dectweb.com/
  http://www.aurelwireless.com/
  http://www.mipot.com/
  http://www.aurel.it

- Sites relacionados com a câmara do emissor

  http://www.geocities.com/siliconvalley/park/1302/gbcam.html
  http://www.lh.co.nz/resources/ar/techinfo.htm
  http://www.soundvisioninc.com/howdcw.htm

- Sites relacionados com o GameBoy™

  http://www.devrs.com/gb
  http://www.ziegler.desaign.de
  http://www.bung.com.hk/porducts/xchanger.htm
  http://n64.icequake.net/mirror/www.bung.com.hk/porducts/xchanger.htm

11. Cartaz

universidade de aveiro 30 anos a projectar futuros

ieeta 2003/2004

Intercomunicador de Imagem e Som sem Fios

Alunos: Sérgio Ivan Lopes nº 18238 Fernando Jorge Ventura nº 13246

Orientadores: Prof. Rui Escadas Prof. Armando Pinto

Objectivos:

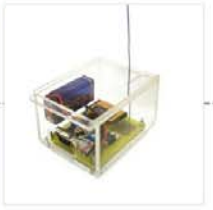
- Projectar o hardware de um intercomunicador de som e imagem sem fios.
- Desenvolver um sistema portátil e de baixo custo, baseado num Gameboy™.
- Implementar um protocolo de comunicações para o sistema .
- Criar software de interface gráfico com o utilizador.

Arquitectura do Sistema:

- Sistema pensado para um ambiente multi-utilizador, em que cada elemento da rede tem uma identidade própria.
- Rede composta por, dois ou mais elementos (Câmaras, Displays, PC's).
- Comunicações sustentadas por um sistema de comunicação local, LAN (Local Area Network) sem fios.
- Gestão das comunicações através do Protocolo implementado, GBDT (GameBoy Data Transfer Protocol).

Protótipo:

- Todos os elementos do sistema são controlados pelo microcontrolador PIC18F458 da Microchip.
- O emissor foi projectado com recurso a um sensor digital de imagem CMOS, o M64282FP, da Mitsubishi.
- O receptor desenvolvido faz o interface com o Gameboy™.
- As comunicações ficam a cargo de um módulo de RF digital, da RADIOMETRIX, half-duplex com uma taxa de débito máxima de 160kbps.
- Foi desenvolvido um interface para o PC com aplicação gráfica.



Especificações técnicas:

- Imagem

Cor	:	Preto e Branco
Resolução	:	4-8bits/pixel
Dimensão	:	128x128 pixels
Frame Rate	:	2 a 5 frames/segundo

- Audio Mono

Resolução	:	8 bits/amostra
Amostragem	:	8KHz

- Comunicações

Tx Transmissão	:	de 57600 até 115200bps
Alcance	:	50m em interiores 200m em exteriores

Resultados e conclusões:

- O emissor cumpre os objectivos, com existência de algum ruído sistemático na aquisição da imagem.
- A comunicação com o PC foi conseguida com sucesso, com a resolução máxima, 8 bits/pixel, a uma taxa de 57600 bits por segundo.
- O GBDTP melhorou significativamente a comunicação com garantias de sincronismo, confidencialidade e fiabilidade.
- O receptor cumpre os objectivos apresentados, para o áudio, e para uma imagem com profundidade de cor de 2 bits por pixel .
- Para resoluções mais baixas (Profundidade de Cor), um pré processamento com um filtro de diethering 2D, melhora significativamente a qualidade da imagem.

Apêndice A - Análise qualitativa de uma imagem 128x128 pixeis

A qualidade do serviço, deverá ser um compromisso, entre a largura de banda disponível no canal e a respectiva qualidade de imagem. Para analisar a imagem necessária, uma abordagem qualitativa. Para isso recorreu-se à manipulação de imagem no matlab, como é mostrado na figura seguinte. As imagens que se seguem servem como referência para a comparação com as diferentes resoluções, e para justificar a escolha da respectiva resolução relativa ao número de bits por pixel. Partiu-se de uma imagem, cuja dimensão é 128*128 pixeis, (foi escolhida esta dimensão, devido ao sensor de imagem que se pretende utilizar na aplicação). As imagens seguintes são apresentadas, sem compressão, e todas apresentam diferentes profundidades de cor:

Resolução	1bit/pixel	2bits/pixel	3bits/pixel	**4bits/pixel**
Nº Cores	2 cores	4cores	8cores	**16cores**
Total=	2048 Bytes	4096 Bytes	6144 Bytes	**8192 Bytes**

Observando a figura, a resolução mais satisfatória, está associada à resolução de 4bits/pixel, a qual corresponde a uma palete de 16 cores. Nestas circunstâncias tem-se uma imagem que ocupa 8192 Bytes, ou mais propriamente 8KBytes. Esta análise fundamenta o objectivo que se tentou alcançar, no decorrer do projecto, já que garante que uma imagem com 4bits/pixel tem qualidade suficiente para o fim esperado.

Apêndice B – Dithering examples

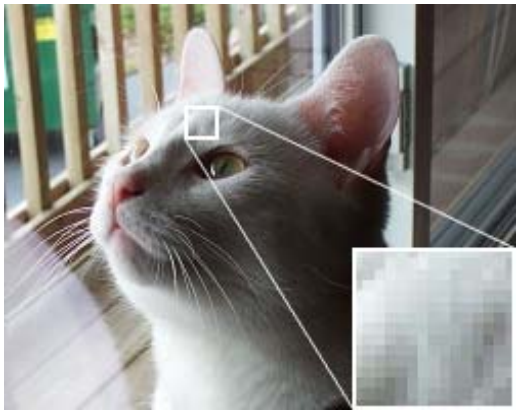


Figure 1. Original photo; note the smoothness in the detail.

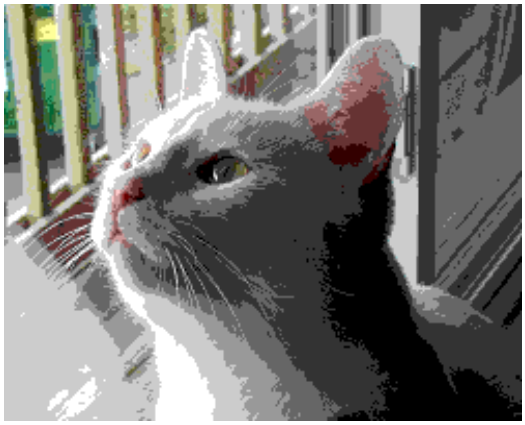


Figure 2. Original image using the web-safe color palette with no dithering applied. Note the large flat areas and loss of detail.

Figure 3. Original image using the web-safe color palette with Floyd-Steinberg dithering. Note that even though the same palette is used, the application of dithering gives a better representation of the original.

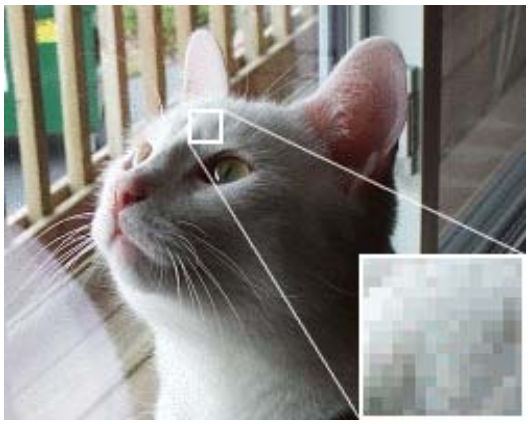


Figure 4. Here, the original has been reduced to a 256-color optimized palette with Floyd-Steinberg dithering applied. The use of an optimized palette, rather than a fixed palette, allows the result to better represent the colors in the original image.

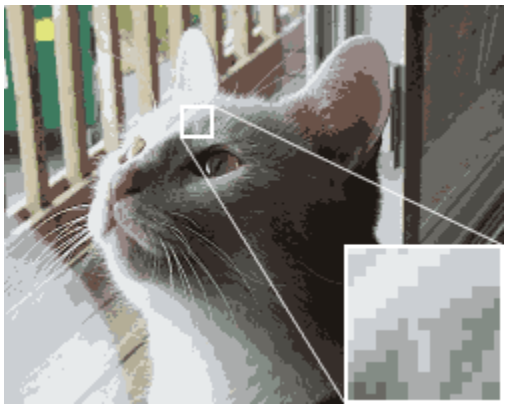


Figure 5. Depth is reduced to a 16-color optimized palette in this image, with no dithering. Colors appear muted, and color banding is pronounced.

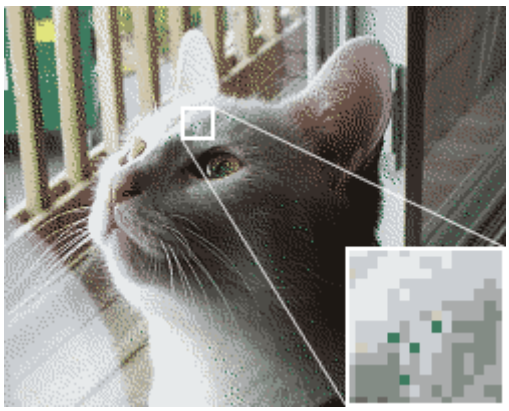
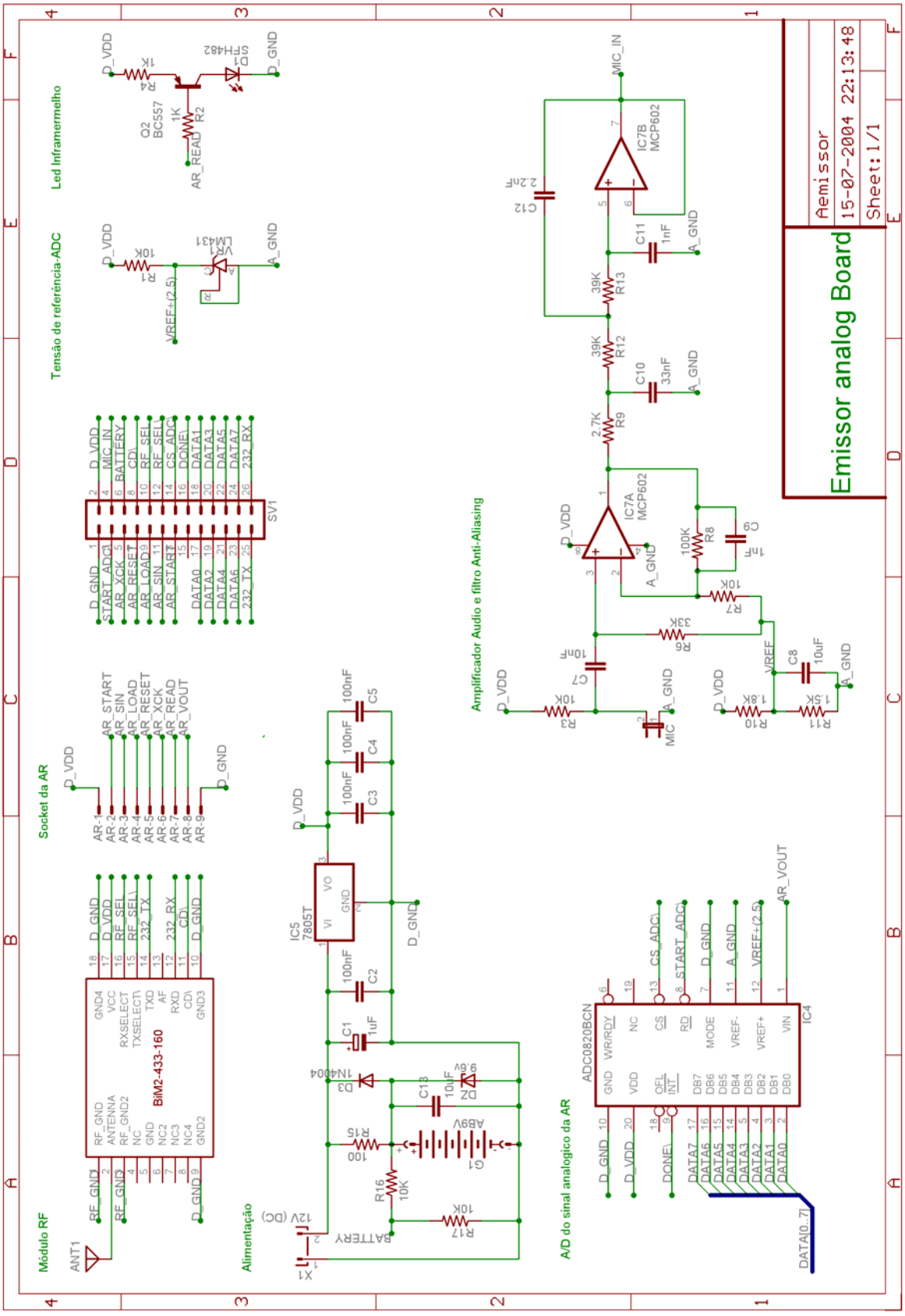
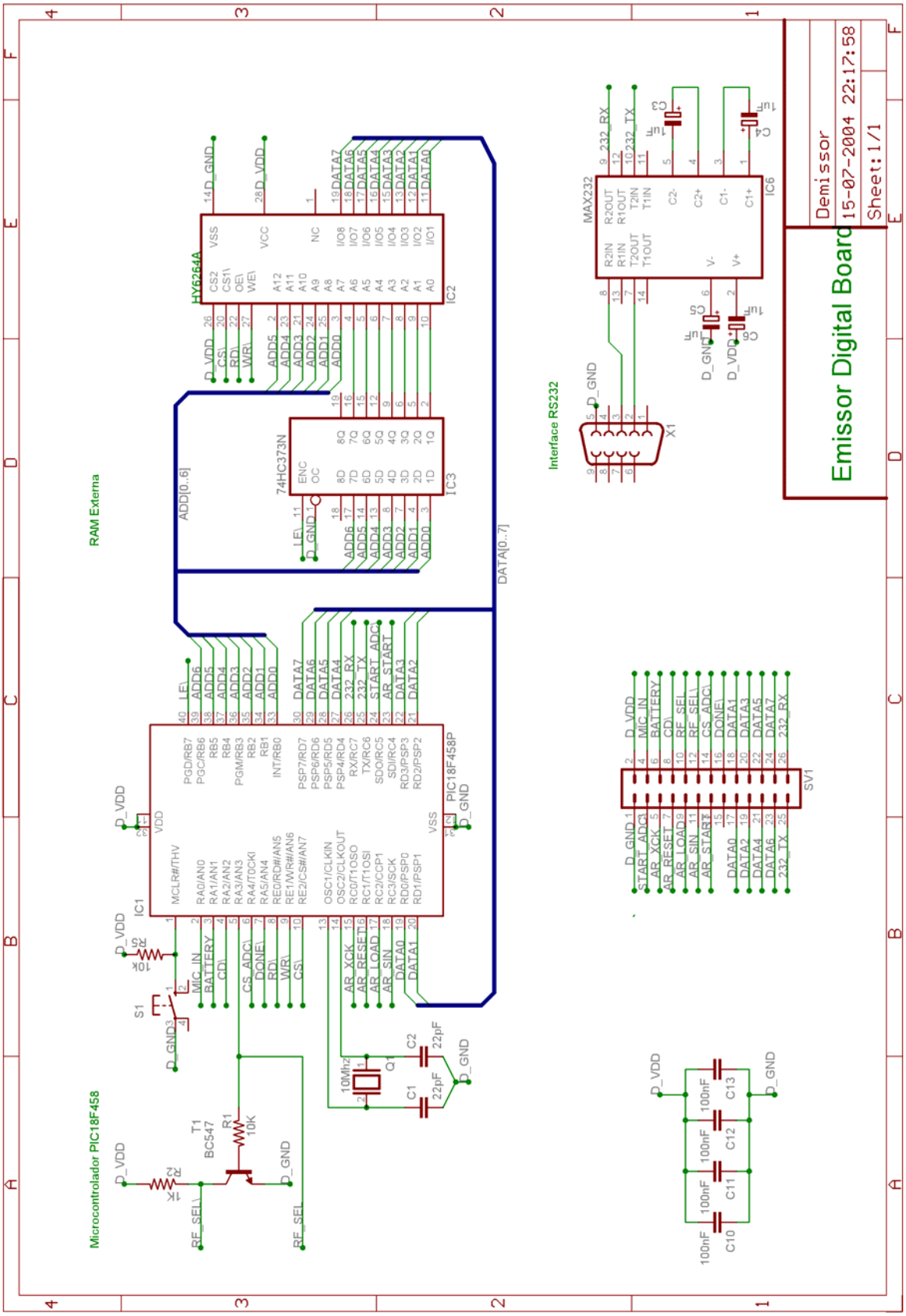
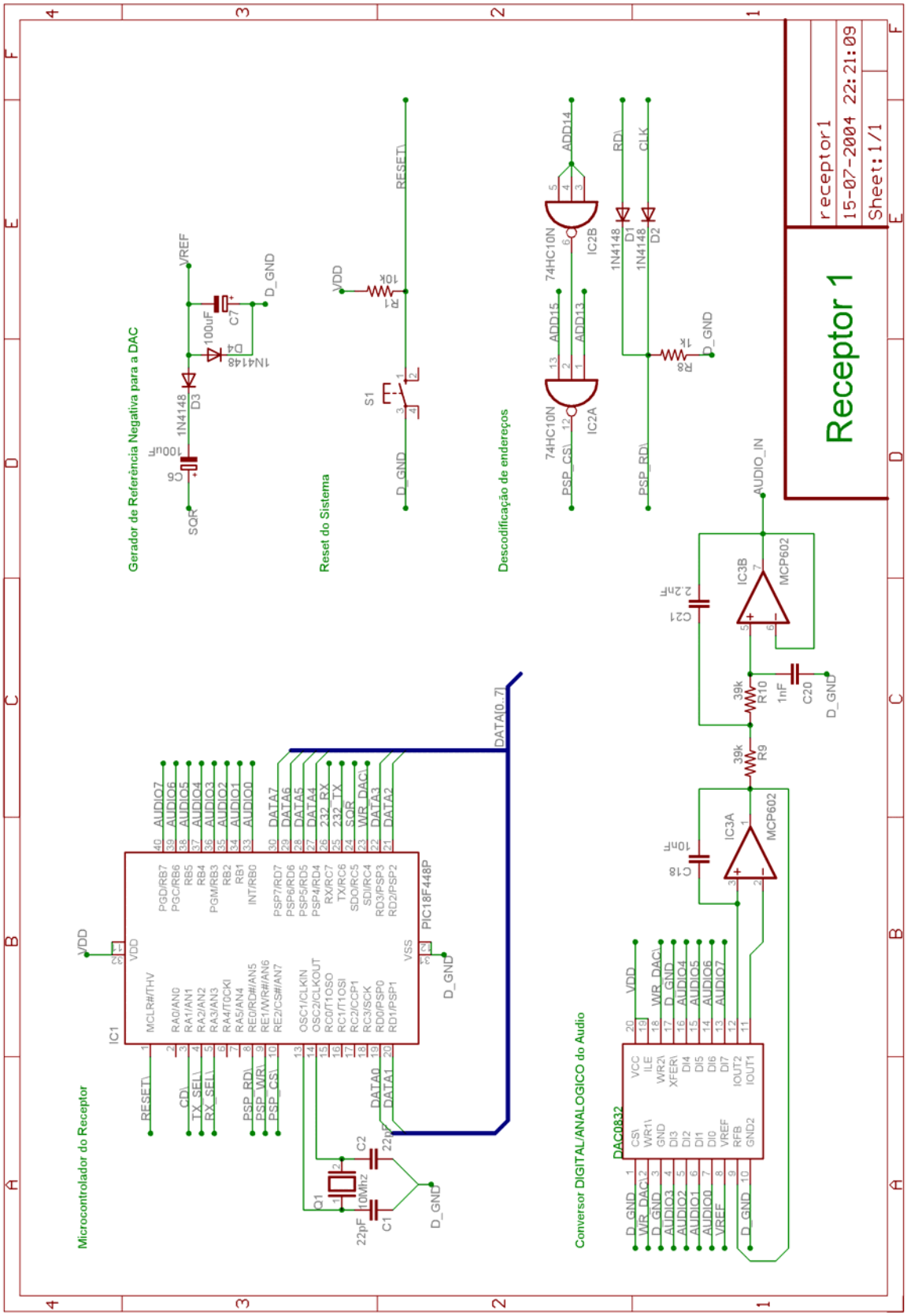


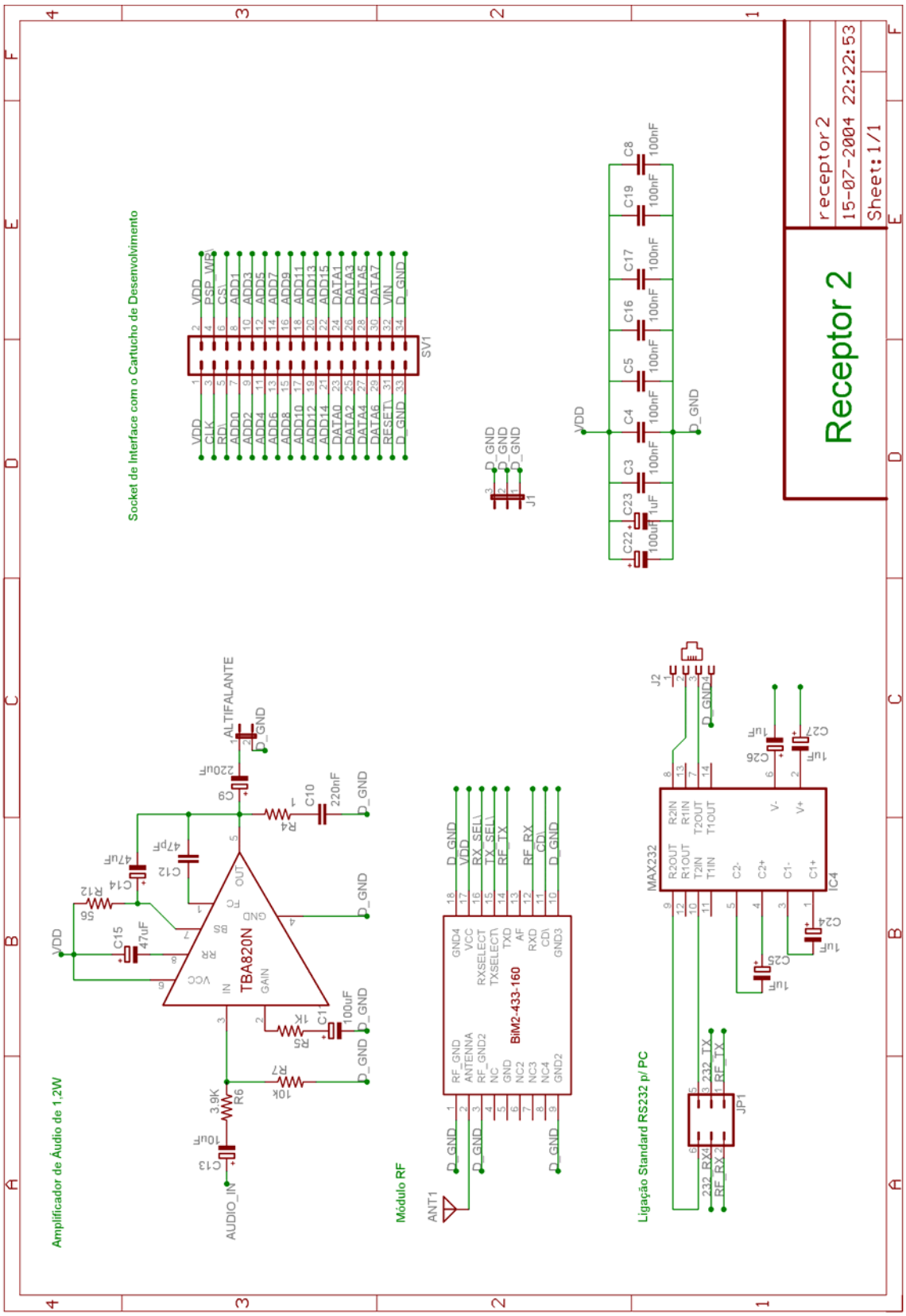
Figure 6. This image also uses the 16-color optimized palette, but the use of dithering helps to reduce banding.

Apêndice C – Esquemas




Emissor Digital Board
Demissor
15-07-2004 22:17:58
Sheet:1/1
RAM Externa
Interface RS232
Microcontrolador PIC18F458
HY6264A
74HC373N
MAX232
PIC18F458P
BC547
ADD[0..6]
DATA[0..7]
D_VDD
D_GND
10Mhz
SV1
X1

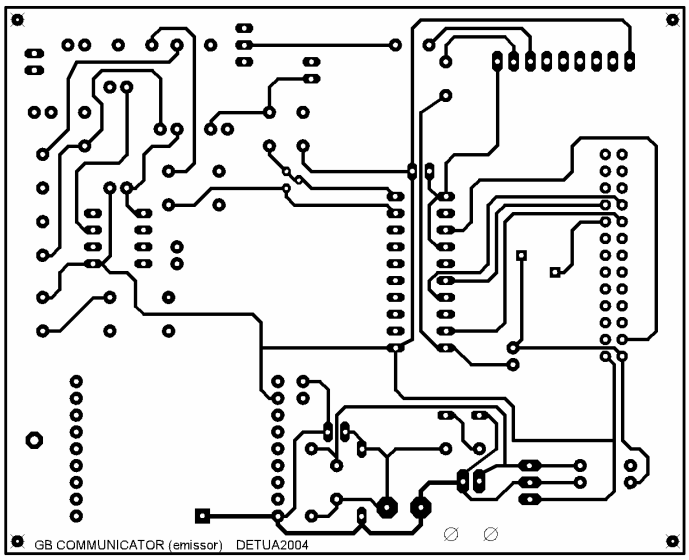

Receptor 1
receptor1
15-07-2004 22:21:09
Sheet:1/1
Gerador de Referência Negativa para a DAC
Reset do Sistema
Descodificação de endereços
Microcontrolador do Receptor
Conversor DIGITAL/ANALOGICO do Audio
PIC18F448P
DAC0832
MCP602
74HC10N
1N4148
VDD
D_GND
RESET\
DATA[0..7]
AUDIO_IN


Receptor 2
receptor2
15-07-2004 22:22:53
Sheet:1/1
Socket de Interface com o Cartucho de Desenvolvimento
SV1
J1
C8 100nF
C19 100nF
C17 100nF
C16 100nF
C5 100nF
C4 100nF
C3 100nF
C23 1uF
C22 100uF
VDD
D_GND
Amplificador de Áudio de 1.2W
TBA820N
ALTIFALANTE
AUDIO_IN
C9 220uF
C10 220nF
C12 47pF
C14 47uF
C15 47uF
C13 10uF
C1 100uF
R4 1
R12 56
R5 1K
R7 10K
R6 3.9K
Módulo RF
BiM2-433-160
ANT1
RF_GND
ANTENNA
RF_GND2
NC
GND
NC2
NC3
NC4
GND2
GND4
VCC
RXSELECT
TXSELECT\
TXD
AF
RXD
CD\
GND3
RX_SEL\
TX_SEL\
RF_TX
RF_RX
Ligação Standard RS232 p/ PC
MAX232
IC4
J2
JP1
C24 1uF
C25 1uF
C26 1uF
C27 1uF
232_TX
RF_TX
232_RX
RF_RX

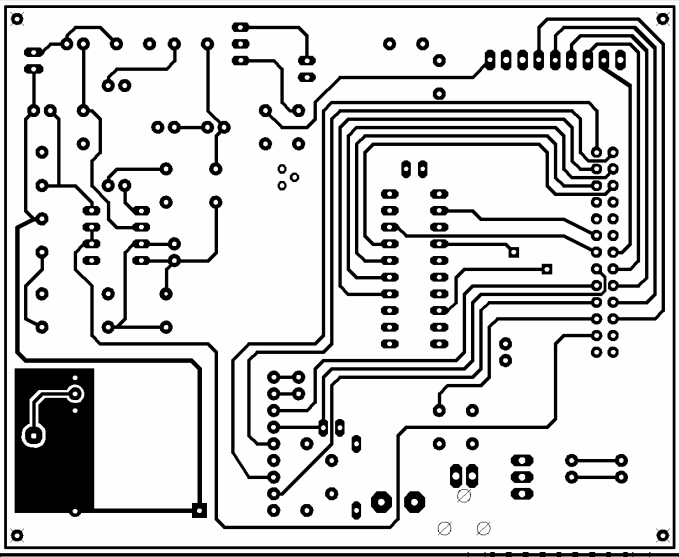
Apêndice D – PCB's

PCB's do EMISSOR

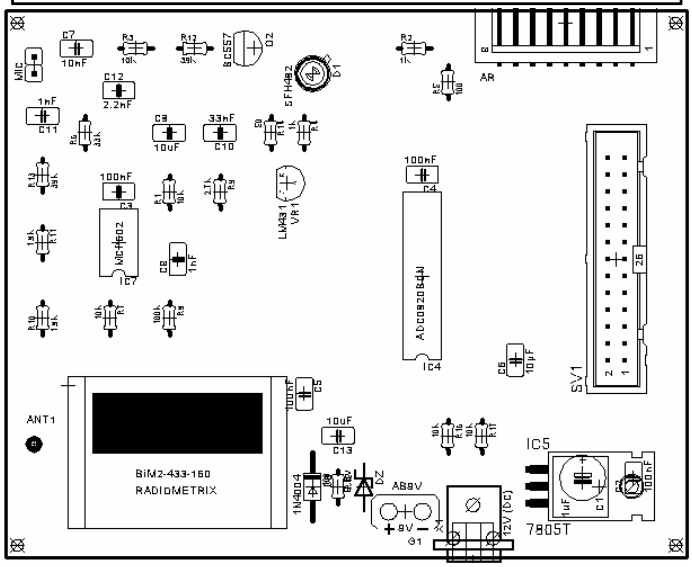
Placa Analógica



Vista de cima

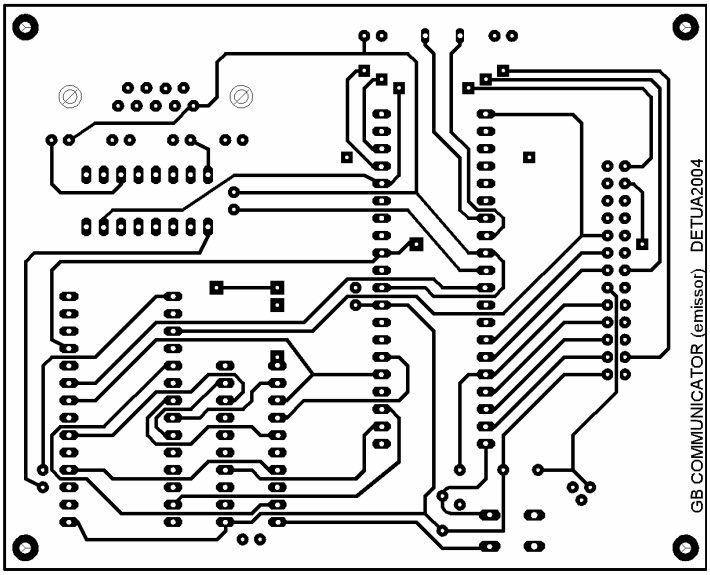


Vista de baixo

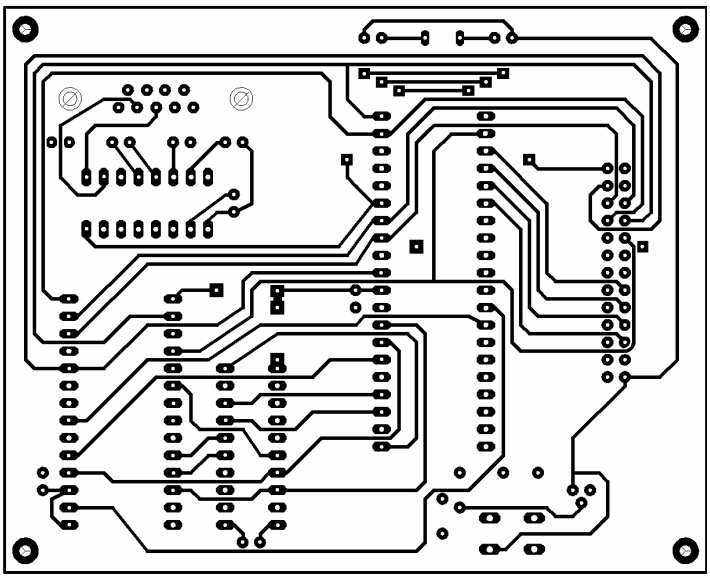


Componentes

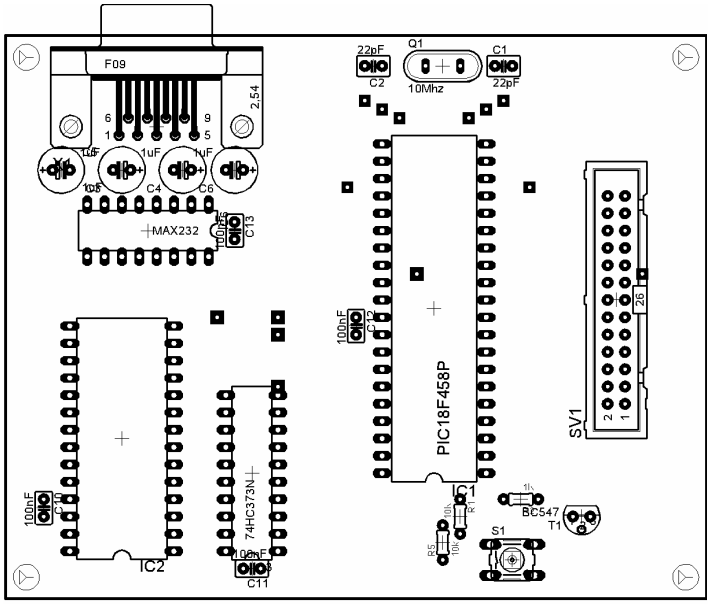
Placa Digital



Vista de cima

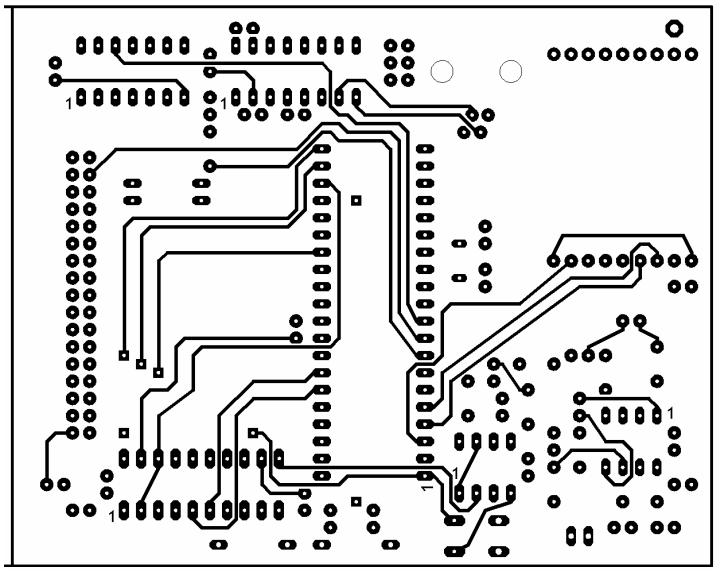


Vista de baixo

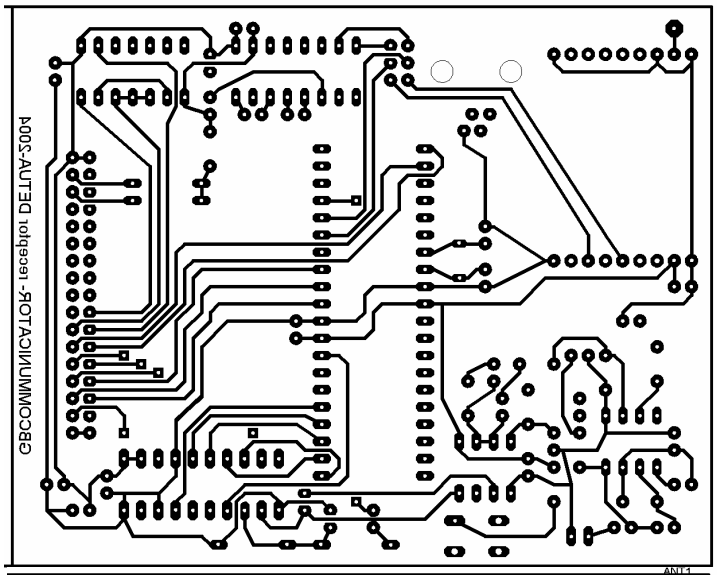


Componentes

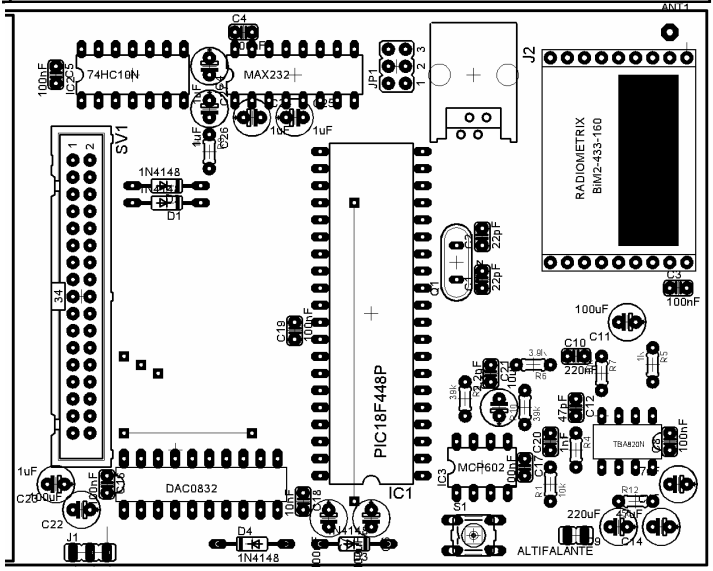
PCB's do RECEPTOR



Vista de cima

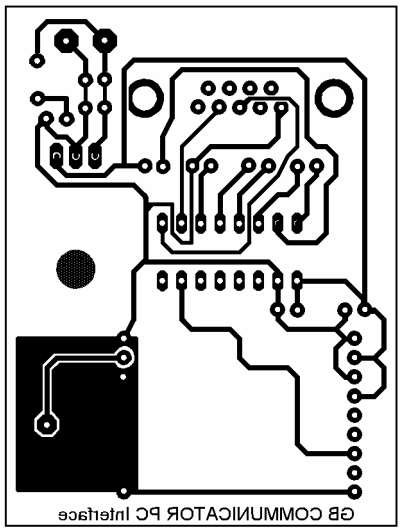


Vista de baixo

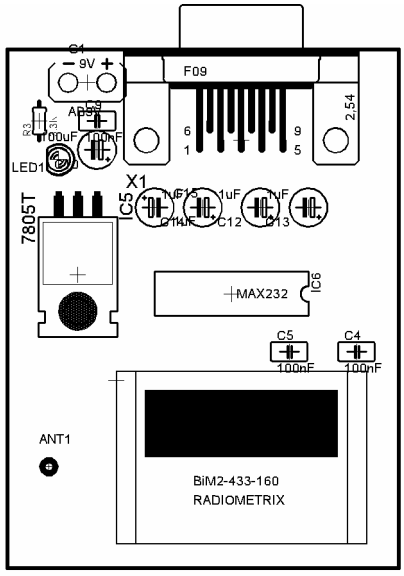


Componentes

PCB do interface para o PC



Vista de baixo



Componentes