

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

DESENVOLVIMENTO DE ECG COM UM CANAL E INTERFACE USB
ATRAVÉS DO MICROCONTROLADOR 8051

ELIEL FREIRE DE MEDEIROS JÚNIOR
GIULER ALBERTO CRUZ SILVA

ORIENTADOR: RICARDO ZELENOVSKY

MONOGRAFIA DE GRADUAÇÃO EM ENGENHARIA ELÉTRICA

PUBLICAÇÃO: ENE 01/2006
BRASÍLIA/DF: AGOSTO – 2006

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

DESENVOLVIMENTO DE ECG COM UM CANAL E INTERFACE USB
ATRAVÉS DO MICROCONTROLADOR 8051

ELIEL FREIRE DE MEDEIROS JÚNIOR
GIULER ALBERTO CRUZ SILVA

MONOGRAFIA SUBMETIDA AO DEPARTAMENTO DE
ENGENHARIA ELÉTRICA DA FACULDADE DE TECNOLOGIA DA
UNIVERSIDADE DE BRASÍLIA COMO PARTE DOS REQUISITOS
NECESSÁRIOS PARA OBTENÇÃO DO GRAU DE ENGENHEIRO
ELETRICISTA.

APROVADA POR:

Prof. Ricardo Zelenovsky, Doutor (ENE-UnB)
(Orientador)

Prof. Geovany Araújo Borges, Doutor (ENE-UnB)
(Examinador)

Prof. Felipe Pereira, Engenheiro (ENE-UnB)
(Examinador)

Humberto Cardoso da Costa, Engenheiro (IME)
(Examinador Externo)

BRASÍLIA/DF, 04 DE AGOSTO DE 2006

FICHA CATALOGRÁFICA

MEDEIROS JÚNIOR, ELIEL FREIRE DE

SILVA, GIULER ALBERTO CRUZ

Desenvolvimento de ECG com um canal e interface USB através do microcontrolador 8051
[Distrito Federal] 2006.

vi, 46 p., 210 x 297 mm (ENE/FT/UnB, Engenheiro Eletricista, 2006)

Monografia de Graduação – Universidade de Brasília. Faculdade de Tecnologia.

Departamento de Engenharia Elétrica.

I. ENE/FT/UnB

II. Título (série)

REFERÊNCIA BIBLIOGRÁFICA

MEDEIROS JÚNIOR, ELIEL FREIRE DE; SILVA, GIULER ALBERTO CRUZ (2006).
Desenvolvimento de ECG com um canal e interface USB através do microcontrolador 8051.
Monografia de Graduação, Publicação ENE 01/2006, Departamento de Engenharia Elétrica,
Universidade de Brasília, Brasília, DF, 04 de agosto de 2006.

CESSÃO DE DIREITOS

AUTORES: Eliel Freire de Medeiros Júnior, Giuler Alberto Cruz Silva.

TÍTULO: Desenvolvimento de ECG com um canal e interface USB através do
microcontrolador 8051.

GRAU: Engenheiro Eletricista

ANO: 2006

É concedida à Universidade de Brasília permissão para reproduzir cópias desta monografia de graduação e para emprestar ou vender tais cópias somente para propósitos acadêmicos e científicos. Os autores reservam outros direitos de publicação e nenhuma parte dessa monografia de graduação pode ser reproduzida sem autorização por escrito dos autores.

Eliel Freire de Medeiros Júnior

Giuler Alberto Cruz Silva

Dedico a minha amada esposa Tânia, a minha filha Maia, e a meus pais Eliel e Eunice.

Eliel Freire de Medeiros Júnior

Dedico a meus pais, Alberto e Ordália, a meus irmãos, Cleide, Geraldo e Rosimeiry e, a minha noiva Julcilene.

Giuler Alberto Cruz Silva

AGRADECIMENTOS

Agradeço,

Primeiramente a Deus, a quem é devida toda honra e toda glória.

A minha esposa, por suportar minha ausência e me apoiar em todas as dificuldades.

A minha filha, por estar sempre presente.

A meus pais, por terem me dado a infra-estrutura para chegar até aqui.

Ao professor Ricardo Zelenovsky, por seu apoio e orientação decisivos.

A Medware, por seu patrocínio.

A todos que de alguma maneira nos apoiaram neste projeto.

Eliei Freire de Medeiros Júnior

Agradeço,

A Deus por me dar forças para superar obstáculos e não deixar me abater.

Aos meus pais e irmãos por compreenderem minha ausência e me apoiarem.

À minha noiva por aceitar minha ausência em diversos momentos e ainda assim me apoiar.

À empresa Medware por patrocinar o projeto.

Ao Engenheiro Humberto Cardoso da Costa pelo auxílio, apoio e incentivo dado a mim para a realização do projeto.

Ao professor Ricardo Zelenovsky que aceitou orientar e que deu total apoio para a realização deste trabalho.

Giuler Alberto Cruz Silva

RESUMO

Desenvolvimento de ECG com um canal e interface USB através do microcontrolador 8051

Autores: Eliel Freire de Medeiros Júnior, Giuler Alberto Cruz Silva

Orientador: Ricardo Zelenovsky

Palavras-chave: USB, ECG, Conversor AD, GLCD, 8051.

Brasília, 04 de Agosto de 2006

O barramento USB se tornou padrão nos microcomputadores pessoais. A elaboração de um dispositivo portátil que utiliza as vantagens desse barramento e a ausência desse tipo de equipamento para alguns tipos de aplicações específicas promoveram a necessidade da elaboração desse trabalho.

Esse projeto visa o domínio da especificação USB, o domínio do projeto e implementação de circuitos para aquisição de sinais de pequena amplitude, mais especificamente os sinais de eletrocardiograma, e o domínio do desenvolvimento de dispositivos microcontrolados para possibilitar em trabalhos futuros o desenvolvimento de um que seja portátil e útil principalmente para locais mais remotos.

Nesse projeto utilizamos uma placa didática “Pentacontroladora” que é dotada de um microcontrolador da família 8051, um conversor analógico-digital modelo ADC12048 da fabricante National Semiconductors, uma ponte USB modelo USBN9603 também da fabricante National Semiconductors, um mostrador gráfico de cristal líquido, memória de 32kB além de outros dispositivos e circuitos que não serão utilizados nesse projeto.

O *Firmware* desenvolvido é responsável por controlar o conversor AD, receber o sinal digitalizado em sua entrada, armazená-lo na memória e, posteriormente plotar esse sinal no mostrador gráfico e enviar para um PC através do barramento USB.

A parte do *firmware* que responde ao tratamento do USB foi adaptada a partir do trabalho desenvolvido por um grupo de estudos anterior. Essa parte deve responder aos pedidos definidos na especificação USB de modo a possibilitar a enumeração da placa junto a um PC tornando-a apta a transmitir e receber dados.

O projeto e a implementação do circuito de aquisição dos sinais cardíacos que compõem o eletrocardiograma foram também adaptados a partir do trabalho desenvolvido por um outro grupo de estudos anterior.

Para o PC poder enviar e receber os dados referentes ao sinal adquirido foi desenvolvido um aplicativo que tem a função de encontrar a placa “Pentacontroladora”, iniciar a transmissão e recepção de dados e possibilitar a plotagem do sinal.

SUMÁRIO

CAPÍTULO I – INTRODUÇÃO	1
1.1 – OBJETIVOS	1
1.2 – MOTIVAÇÃO.....	1
1.3 – ORGANIZAÇÃO DO TRABALHO	2
CAPÍTULO 2 – MICROCONTROLADOR 8051 E A INTERFACE USB	3
2.1 – UMA BREVE DESCRIÇÃO DO BARRAMENTO USB	3
2.1.1 – Características Gerais.....	3
2.1.2 – O processo de Enumeração.....	7
2.1.3 – Tipos de Descritores.....	9
2.2 – A PONTE USBN9603.....	11
2.2.1 – Descrição Funcional	11
2.2.2 – A Solução Adotada para a Transmissão dos Sinais ECG	14
CAPÍTULO 3 – ELETROCARDIOGRAMA (ECG)	15
3.1 – A FISILOGIA DO CORAÇÃO	15
3.2 – PROJETO DO CIRCUITO ANALÓGICO.....	16
3.2.1 – Módulo Amplificador Diferencial.....	18
3.2.2 – Módulo Filtro Passa-altas.....	19
3.2.3 – Módulo Amplificador + Filtro Passa Baixas	20
3.2.4 – Módulo Referência da Perna Direita	21
3.2.5 – Módulo de Gerador de Tensões de Alimentação e Terra	22
CAPÍTULO 4 – CIRCUITOS ADICIONAIS.....	23
4.1 – O CONVERSOR ADC12048.....	23
4.1.1 – Descrição Funcional	23
4.1.2 – A Solução Adotada para a Aquisição dos Sinais ECG.....	27
4.2 – O MOSTRADOR GRÁFICO DE CRISTAL LÍQUIDO (GLCD) AGM-12864K	27
4.2.1 – Descrição Funcional	27
4.2.2 – A Solução Adotada para Plotar os Sinais ECG.....	31
4.3 – A MEMÓRIA EXTERNA 62256	31
4.3.1 – Descrição Funcional	31
4.3.2 – A Solução Adotada para Armazenar os Sinais ECG.....	32
CAPÍTULO 5 – PROGRAMAS	34
5.1 – FIRMWARE.....	34
5.1.1 – Laço Principal.....	34
5.1.2 – Aquisição, Conversão AD e Armazenagem	35

5.1.3 – Enumeração, Transmissão e Recepção USB.....	37
5.1.4 – Plotagem no GLCD	39
5.2 – APLICATIVO WINDOWS	40
CAPÍTULO 6 – RESULTADOS.....	42
6.1 – ELETROCARDIOGRAMA (ECG)	42
6.2 – DIGITALIZAÇÃO DOS SINAIS	43
CAPÍTULO 7 – CONCLUSÃO.....	46
REFERÊNCIAS BIBLIOGRÁFICAS	48

LISTA DE TABELAS

TABELA 2.1: CONDUTORES QUE COMPÕES UM CABO USB.....	5
TABELA 2.2: TIPOS DE TRANSAÇÃO DO PADRÃO USB [1].....	6
TABELA 4.1: DESCRIÇÃO DOS <i>BITS</i> DO REGISTRADOR DE CONFIGURAÇÃO DO ADC12048 [11].....	24
TABELA 4.2: COMANDOS DISPONÍVEIS AO CONVERSOR ADC12048 [11].....	24
TABELA 4.3: DESCRIÇÃO DOS <i>BITS</i> DO REGISTRADOR DE DADOS DO ADC12048 [11].....	26
TABELA 4.4: OPERAÇÕES DISPONÍVEIS NO MOSTRADOR GRÁFICO AGM-12864K	29
TABELA 4.5: INSTRUÇÕES DISPONÍVEIS NO MOSTRADOR GRÁFICO AGM-12864K [7]	30
TABELA 5.1: COMANDOS UTILIZADOS NA COMUNICAÇÃO PC – “PENTACONTROLADORA”	34
TABELA 5.2: DESCRIÇÃO DOS ESTADOS INFORMADOS PELOS LEDS.....	37

LISTA DE FIGURAS

FIGURA 2.1: DIAGRAMA DE BLOCOS DO USBN9603 [12]	11
FIGURA 2.2: HIERARQUIA DOS REGISTRADORES DO USBN9603 [12]	13
FIGURA 3.1: ONDAS P, Q, R, S, T E U EM UM TRAÇADO DE ECG.....	15
FIGURA 3.2: ESQUEMÁTICO DOS MÓDULOS PROJETADOS PARA O CIRCUITO DE ECG.....	17
FIGURA 3.3: CIRCUITO PARA MEDIÇÃO E CONDICIONAMENTO DOS SINAIS DE ECG.....	17
FIGURA 3.4: CIRCUITO MÓDULO AMPLIFICADOR DIFERENCIAL	18
FIGURA 3.5: CIRCUITO MÓDULO FILTRO PASSA ALTAS	19
FIGURA 3.6: CIRCUITO MÓDULO AMPLIFICADOR + FILTRO PASSA BAIXAS.....	21
FIGURA 3.7: CIRCUITO MÓDULO DE REFERÊNCIA DA PERNA DIREITA	22
FIGURA 3.8: CIRCUITO MÓDULO GERADOR DE TENSÕES DE ALIMENTAÇÃO E TERRA	22
FIGURA 4.1: DIAGRAMA DE BLOCOS DO ADC12048 [11].....	23
FIGURA 4.2: DIAGRAMA DE BLOCOS DA ALIMENTAÇÃO DO GLCD AGM-12864K [9].....	28
FIGURA 4.3: DIAGRAMA DE BLOCOS DO CIRCUITO LÓGICO DO GLCD AGM-12864K [9].....	28
FIGURA 4.4: COMPOSIÇÃO DA MATRIZ 64 X 64 DO GLCD AGM-12864K [7].....	29
FIGURA 4.5: DIAGRAMA DE BLOCOS DA MEMÓRIA 62256 [10]	32
FIGURA 5.1: FLUXOGRAMA BÁSICO DO LAÇO PRINCIPAL DO <i>FIRMWARE</i>	35
FIGURA 5.2: FLUXOGRAMA BÁSICO PARA PARTE USB DO <i>FIRMWARE</i> [4]	37
FIGURA 5.3: TELA DO PROGRAMA APLICATIVO.....	41
FIGURA 6.1: SINAL DE ECG OBTIDO NO OSCILOSCÓPIO	42
FIGURA 6.2: SINAL DE ECG OBTIDO NO OSCILOSCÓPIO COM JANELA DE TEMPO MAIOR.....	43
FIGURA 6.3: RESULTADO DA DIGITALIZAÇÃO DE UM SINAL SENOIDAL SIMÉTRICO	44
FIGURA 6.4: RESULTADO DA DIGITALIZAÇÃO DE UM SINAL SENOIDAL ASSIMÉTRICO.....	44
FIGURA 6.5: RESULTADO DA DIGITALIZAÇÃO DE UM SINAL TRIANGULAR	44
FIGURA 6.6: RESULTADO DA DIGITALIZAÇÃO DE UMA ONDA QUADRADA.....	44
FIGURA 6.7: RESULTADO DA DIGITALIZAÇÃO DO SINAL DE ECG.....	44

CAPÍTULO I – INTRODUÇÃO

1.1 – OBJETIVOS

Esse trabalho propõe o projeto de um circuito analógico de eletrocardiograma com um canal para ser ligado ao conversor analógico-digital da placa didática “Pentacontroladora” e depois do sinal convertido para digital ser armazenado em memória, mostrado na tela gráfica de cristal líquido presente na placa e enviado para um PC a partir do barramento USB. A placa didática “Pentacontroladora” faz uso do microcontrolador 8051, da ponte USB USBN9603 da fabricante National que possibilita a comunicação da placa com o barramento USB do PC; de um conversor analógico digital ADC12048 com resolução de 12 bits e 8 canais analógicos também da fabricante National; de um módulo de memória RAM com 32kB de capacidade e de um mostrador gráfico de cristal líquido modelo AGM-12864K da fabricante AGTechnologies.

1.2 – MOTIVAÇÃO

O domínio da tecnologia USB (*Universal Serial Bus*) que está em ascensão no mercado possibilita o projeto e a montagem de dispositivos com alta performance de transferência de dados e uma grande versatilidade. A dificuldade do desenvolvimento de tais dispositivos e a ausência de informações detalhadas sobre a tecnologia tornam o mercado escasso de profissionais o que nos motiva a pesquisar e desenvolver este trabalho.

O domínio dos circuitos de aquisição de sinais biomédicos associado com os modernos dispositivos disponíveis no mercado e os custos relativamente baixos possibilitam o desenvolvimento de projetos portáteis e que facilitem a vida dos pacientes e dos médicos permitindo inclusive que um monitor cardíaco, por exemplo, esteja acessível na residência de um paciente.

O uso da placa didática “Pentacontroladora” e seus muitos recursos possibilita a prática dos conhecimentos teóricos obtidos em diversas disciplinas do curso, mais especificamente os conhecimentos em eletrônica digital e em processadores digitais.

A possibilidade da continuidade do trabalho através da miniaturização do dispositivo a partir do projeto de hardware específico e a possibilidade deste se tornar um produto comercial também serve de motivação para o desenvolvimento desse trabalho.

1.3 – ORGANIZAÇÃO DO TRABALHO

O trabalho está organizado em sete capítulos.

O segundo capítulo trata da solução adotada na placa didática “Pentacontroladora” para o interfaceamento entre o microcontrolador 8051 e a ponte USBN9603 e desta para o *host* USB no PC.

O terceiro capítulo mostra um breve resumo teórico sobre o eletrocardiograma (ECG), detalhando a fisiologia do coração, a amplitude e o formato das ondas esperadas e termina mostrando o projeto do circuito analógico utilizado para captar, amplificar e filtrar os sinais cardíacos.

O quarto capítulo apresenta as características dos circuitos e dispositivos utilizados para a conversão analógica-digital, armazenamento dos sinais na memória RAM e a plotagem dos mesmos no mostrador gráfico de cristal líquido.

O quinto capítulo apresenta a descrição funcional da solução de software adotada para interligar todos os dispositivos e colocar em funcionamento a aquisição, a conversão, a armazenagem e a apresentação do ECG.

O sexto capítulo apresenta os resultados obtidos no projeto tanto da parte analógica que foi medida com osciloscópio quanto dos dados digitalizados plotados pelo programa desenvolvido.

O sétimo capítulo apresenta a conclusão do trabalho, a confirmação dos resultados esperados e a possibilidade trabalhos futuros.

CAPÍTULO 2 – MICROCONTROLADOR 8051 E A INTERFACE USB

2.1 – UMA BREVE DESCRIÇÃO DO BARRAMENTO USB

2.1.1 – Características Gerais

O barramento *Universal Serial Bus* (USB) tem, desde seu lançamento, se tornado um padrão na microinformática. O seu desenvolvimento partiu da colaboração de diversas empresas líderes de mercado nos segmentos de software e hardware. A sua popularidade se deve dentre outras características:

- à facilidade do uso: na grande maioria dos casos não há a necessidade de configurações no hardware para que o mesmo funcione, basta conectá-lo a um *host* e o mesmo se encarrega de reconhecer a conexão elétrica do dispositivo.
- à velocidade: o barramento não se torna o gargalo da comunicação entre o dispositivo e um microcomputador para o caso em questão, por exemplo.
- à confiabilidade: os erros são raros de ocorrer e quando ocorrem são automaticamente corrigidos pelos algoritmos implantados com o padrão.
- à flexibilidade: uma grande variedade de periféricos e dispositivos pode ser utilizada utilizando para comunicação o padrão USB.
- ao preço: os custos baixos dos componentes não são empecilhos para os fabricantes construírem a interface com seus produtos e criam uma vantagem para os consumidores finais.
- ao suporte dado pelos sistemas operacionais dos microcomputadores: os desenvolvedores, na maioria das vezes, não necessitam de escrever *drivers* de baixo nível para os periféricos que utilizam o barramento.

O padrão se mostra bastante fácil para os usuários finais dos produtos que o utilizam para comunicação. Essa única interface permite, com diferentes tipos de conectores também padronizados, a utilização de até 127 dispositivos diferentes através da ligação de *hubs*. O processo automático de configuração permite que ao conectar o dispositivo ou periférico no barramento o *hub* detecta o dispositivo e faz uma ponte com o *host* que comunica ao sistema

operacional e este já providencia a configuração, com isso não há a necessidade de reiniciar o sistema antes de usar o dispositivo. Como cada dispositivo compartilha o barramento, os recursos do sistema ficam livres para outros periféricos que não possuam essa facilidade.

O USB suporta barramentos com três velocidades de acordo com a versão da especificação. A versão 1.0 (*low-speed*) atinge a taxa de 1,5Mbps. A versão 1.1 (*full-speed*) atinge a taxa de 12Mbps. Por fim, a versão 2.0 (*high-speed*) atinge a taxa de 480Mbps. Todo microcomputador compatível com o USB opera por padrão nas versões 1.0 e 1.1. A versão 2.0 da especificação requer um *hardware* específico presente na placa-mãe do microcomputador ou em uma placa de expansão.

Os principais sistemas operacionais utilizados nos microcomputadores têm suporte ao padrão USB. Na família PC os sistemas operacionais gratuitos com código fonte aberto da família Linux possuem suporte ao barramento bem como a família de sistemas operacionais comerciais Windows da fabricante Microsoft. Na família Apple de microcomputadores todos os dispositivos são conectados através do barramento USB e o sistema operacional oferece o suporte ao funcionamento. O suporte oferecido pelos sistemas operacionais consiste em detectar quando um dispositivo é conectado ou desconectado do sistema, comunicar com os dispositivos recém conectados para saber a velocidade, o modo de operação e a quantidade de dados a ser transmitido e, prover um mecanismo que permita os programas de *drivers* a se comunicarem com o hardware controlador *host*.

A desenvolvedora de *softwares* Microsoft disponibilizou na sua família de sistemas operacionais Windows uma série de pacotes de *drivers* para diversos dispositivos. Há pacotes para dispositivos de áudio, modems, câmeras, *scanners*, impressoras, dispositivos de armazenamento e para dispositivos de interação humana como teclados, mouses e controles para jogos. Esse último pacote é conhecido como HID (*human interface devices*) e facilita muito o desenvolvimento de dispositivos devido a sua simplicidade comparada às demais classes.

Cada dispositivo que utiliza o barramento USB deve possuir um microcontrolador para manipular o protocolo de comunicação que tem um grau relativamente elevado de complexidade. Todavia, alguns fabricantes de circuitos integrados desenvolveram dispositivos conhecidos como pontes que implementam o protocolo de comunicação bem como a geração dos sinais e facilitam o desenvolvimento de *hardwares* uma vez que a montagem dos pacotes com o rigor que o padrão exige é realizada por estes dispositivos.

O USB foi projetado para operar com comunicação serial assíncrona, ou seja, sem o uso de um sinal de relógio de referência e com sinais de dados diferenciais. Para manter o

sincronismo dos sinais entre transmissor e receptor é utilizada a codificação NRZI (*Non-Return to Zero Inverted*). O comprimento máximo do cabo de comunicação é de cinco metros. O cabo é blindado e composto por condutores conforme mostrado na tabela 2.1.

Pino	Sinal	Cor
1	VBus (+5V)	Vermelho
2	D-	Branco
3	D+	Verde
4	GND	Preto

TABELA 2.1: CONDUTORES QUE COMPÕES UM CABO USB

Cada dispositivo USB, pela especificação, deve implementar subdispositivos que respondam eletricamente às transações. Esses subdispositivos são chamados de *Endpoints*. Cada *Endpoint* recebe um endereço único que é composto por um número e pela direção dos dados, se envia ou recebe. Todo dispositivo deve ter um *Endpoint* de número zero que realiza transações de controle e portanto envia e recebe dados. Na verdade o *Endpoint* de número zero é composto por dois *Endpoints*, um para envio e outro para recepção, no mesmo endereço. São disponibilizados quatro bits para endereçamento dos *Endpoints* permitindo, assim, um total de dezesseis incluindo o de número zero.

A transferência de dados no padrão USB é realizada através de pacotes de dados. Os pacotes são do tipo *token*, *data* e *handshake*.

Os principais pacotes do tipo *token* são os pacotes *setup* que são utilizados para indicar que o próximo pacote *data* será para escrita no *endpoint* zero; pacotes *in* que são utilizados para indicar que o próximo pacote *data* deverá ser lido do *endpoint* de saída e; pacotes *out* que indicam que o próximo pacote *data* deverá ser escrito no *endpoint* de entrada.

Os pacotes do tipo *data* podem ser *data1* e *data0* e são efetivamente os dados transportados do *host* para o dispositivo ou do dispositivo para o *host* de acordo com o pacote *token* que os precederam.

Os pacotes do tipo *handshake* podem ser *ack*, *nak* e *stall*. O *ack* é utilizado para informar ao receptor, *host* ou dispositivo, que o dado foi recebido corretamente. O *nak* é utilizado para informar que o *host* ou dispositivo está ocupado ou inoperante no momento. Por fim o *stall* é utilizado para indicar um erro na comunicação.

A comunicação entre um dispositivo USB e um *host* é realizada mediante transações. A transação nada mais é que um conjunto de pacotes enviados e recebidos sem erros. Podem

ser de quatro tipos: *control*, *bulk*, *interrupt* e *isochronous*. As transações do tipo *control* devem ser implementadas por todos os dispositivos USB e têm a finalidade de permitir ao *host* o envio de solicitações de informações sobre o dispositivo e ler essas informações. As transações do tipo *bulk* podem ser implementadas apenas nos dispositivos que operam nas versões 1.1 (*full-speed*) e 2.0 (*high-speed*) e são utilizadas quando a taxa de transferência de dados não é crítica, isto é, o dispositivo ou o *host* podem aguardar a liberação do barramento quando este está ocupado. As transações do tipo *interrupt* podem ser implementadas por qualquer dispositivo USB e são utilizadas quando o dispositivo ou o *host* necessitam de atenção periodicamente como um mouse ou um teclado, por exemplo. As transações do tipo *isochronous* podem ser implementadas apenas em dispositivos que operam nas versões 1.1 (*full-speed*) e 2.0 (*high-speed*) e são utilizadas por dispositivos que necessitam da garantia do tempo de entrega dos dados mas que podem tolerar erros ocasionais uma vez que nesse tipo de transação não há a retransmissão de dados. A tabela 2.2 resume as características dos tipos de transação.

Tipo de transação	<i>Control</i>	<i>Bulk</i>	<i>Interrupt</i>	<i>Isochronous</i>
Obrigatoriedade	Sim	Não	Não	Não
Versões que dão suporte	Todas	1.1 e 2.0	Todas	1.1 e 2.0
Direção do fluxo de dados	Entrada e saída	Entrada ou saída	Entrada ou saída (a versão 1.0 só suporta entrada)	Entrada ou saída
Correção de erros	Sim	Sim	Sim	Não
Taxa de transferência fixa	Não	Não	Não	Sim
Tipos de pacotes utilizados	<i>Token, Data e Handshake</i>	<i>Token, Data e Handshake</i>	<i>Token, Data e Handshake</i>	<i>Token e Data</i>
Tempo máximo entre transferências	Não	Não	Sim	Sim
Uso típico	Configuração	Impressoras, scanners	Mouses, teclados	Áudio

TABELA 2.2: TIPOS DE TRANSAÇÃO DO PADRÃO USB [1]

2.1.2 – O processo de Enumeração

A enumeração é a troca inicial de informações entre o dispositivo e o *host* cuja finalidade é identificar o dispositivo e atribuir a este um *device driver* que possibilita a comunicação com as aplicações.

O processo de enumeração é realizado através de transações do tipo *control* no *Endpoint 0* e inclui a atribuição de um endereço ao dispositivo, a leitura de estruturas de dados que caracterizam o dispositivo, a atribuição e a carga de um *device driver* e a seleção de configurações de acordo com os dados recebidos que incluem o tipo de transações e os *endpoints* que serão utilizados.

Uma das etapas do processo de enumeração é a detecção da conexão ou remoção do dispositivo no barramento. Quando o sistema inicializa é realizada uma varredura no barramento para detectar os dispositivos já conectados e depois que o sistema já está inicializado são realizadas varreduras periódicas para verificar se houve alguma alteração no barramento como uma conexão ou remoção de dispositivos.

Quando um dispositivo é detectado o *hub* comunica ao *host* e este envia uma série de requisições e comandos ao dispositivo. Para que esse dispositivo seja enumerado ele deve responder a cada requisição com os dados solicitados e realizar ações comandadas pelo *host*.

A ideia do processo é que ele seja transparente ao usuário final e ocorra de maneira automática. No ambiente Windows apenas uma mensagem é mostrada e dependendo do tipo do dispositivo é necessário inserir um disco de *drivers* na primeira vez que a conexão ocorre.

Os passos do processo de enumeração são os seguintes:

- O dispositivo é conectado a uma porta USB: o *hub* fornece alimentação ao dispositivo e ele passa ao estado alimentado.
- O *hub* detecta o dispositivo: as tensões nas linhas de sinal são monitoradas a partir de dois resistores de *pull-down* no *hub*. Quando há uma mudança ocasionada por um resistor de *pull-up* no lado do dispositivo na linha D+ é reconhecido um dispositivo versão 1.1 (*full-speed*) ou 2.0 (*high-speed*). Quando o resistor de *pull-up* no lado do dispositivo está na linha D- é reconhecido um dispositivo versão 1.0 (*low-speed*). Nessa fase, no entanto, a versão do dispositivo não é detectada, apenas a mudança no estado das linhas de sinal.

- O *host* reconhece o novo dispositivo: o *hub* raiz relata um evento ao *host* e repassa em qual porta e/ou outro *hub*, quando for o caso, ocorreu a mudança de estado.
- O *hub* detecta a versão USB suportada pelo dispositivo: antes do dispositivo ser iniciado o *hub* detecta a versão USB e portanto a velocidade do dispositivo e informa ao *host*.
- O *hub* inicializa o dispositivo: o *host* envia um comando ao *hub* para iniciar a porta onde foi detectada a presença do novo dispositivo. O *hub* então coloca nas linhas de dados a condição de *reset* que é percebida apenas pelo dispositivo e dura em média 10ms.
- O *host* verifica se o dispositivo da versão 1.1 (*full-speed*) suporta a versão 2.0 (*high-speed*): alguns sinais especiais são enviados ao dispositivo e se este consegue interpretar e responder a esses sinais todas as próximas transações são assumidas na versão 2.0 (*high-speed*).
- O *hub* estabelece um caminho entre o dispositivo e o barramento: o *host* consulta o *hub* se o dispositivo já saiu da condição de *reset*. Essa consulta é realizada até que a resposta seja positiva. Quando o dispositivo sai da condição de *reset* ele está pronto a se comunicar a partir do *Endpoint 0* e transações de controle com o *host*. Nessa etapa o dispositivo pode consumir até 100mA de corrente.
- O *host* consulta o tamanho máximo do pacote utilizado pelo dispositivo: o *host* envia uma requisição ao dispositivo e aguarda um descritor contendo o tamanho máximo do pacote utilizado pelo *Endpoint 0* que está presente no oitavo *byte* do descritor.
- O *host* atribui um endereço ao dispositivo: o *host* envia um comando ao dispositivo indicando a este qual é o seu endereço enquanto este estiver conectado no barramento. O dispositivo envia uma confirmação ao *host* e armazena internamente seu endereço atribuído.
- O *host* identifica as funcionalidades do dispositivo: o *host* solicita, através do endereço atribuído, informações de quais funcionalidades são disponibilizadas e o dispositivo responde enviando um descritor de dispositivo que nada mais é que uma estrutura de dados padronizada. O *host* solicita então os descritores de configuração que indicam informações complementares como quais *Endpoints* serão utilizados e quais os seus tipos, tamanho dos pacotes, etc. O dispositivo responde entregando os descritores de configuração solicitados.

- O *host* atribui e carrega um *Device Driver*: o *host* com base nos dados de fabricante e produto recebidos pelo descritor de dispositivo repassa ao sistema operacional qual *Device Driver* foi atribuído ao dispositivo e solicita a sua carga em memória.
- O *Device Driver* seleciona as configurações: o *host* envia comandos de configuração ao dispositivo de acordo com a solicitação do *Device Driver*. Nesse ponto o dispositivo está enumerado e pronto para o uso.

A remoção do dispositivo é detectada pelo *hub* que desliga a porta do dispositivo e repassa ao *host* o ocorrido. Este, por sua vez, informa ao sistema operacional que o dispositivo não está mais disponível, solicita a descarga do *Device Driver* e libera o endereço utilizado.

2.1.3 – Tipos de Descritores

Como visto no processo de enumeração, os descritores são as estruturas de dados, ou seja, blocos de informação formatados que são enviados do dispositivo ao *host* como uma resposta a uma requisição deste.

À medida que o processo de enumeração ocorre, os descritores são solicitados na ordem incremental de detalhes, isto é, começa com descritores que detalham o dispositivo de maneira geral e vai passando a descritores que detalham cada configuração, depois cada interface de configuração e por fim cada interface de *Endpoint*.

Cada dispositivo deve possuir um e apenas um descritor de dispositivo que contém informações sobre o dispositivo e especifica o número de configurações suportadas por este. Cada dispositivo tem um ou mais descritores de configuração que contém informações como por exemplo o consumo de potência e o número de interfaces de configuração. Cada descritor de interface pode referenciar mais que um descritor de *Endpoint* que contém as informações necessárias para realizar a comunicação com o *Endpoint*. Uma interface que não possui descritor de *Endpoint* utiliza o *Endpoint* de controle para comunicação.

Os descritores podem ser dos seguintes tipos:

- Descritor de Dispositivo: esse descritor é obrigatório para qualquer dispositivo. Ele contém informações básicas sobre o dispositivo, é o primeiro a ser solicitado e possibilita ao *host* as informações necessárias para solicitar as informações adicionais do dispositivo. Esse descritor informa a versão USB suportada pelo dispositivo, qual a

classe, subclasse, produto e fabricante o dispositivo faz parte, qual o tamanho máximo dos pacotes utilizados no *Endpoint 0*, qual a quantidade de configurações possíveis e informa também quais os descritores de *String* são relacionados ao fabricante, descrição do produto e número de série.

- **Descritor Qualificador de Dispositivo:** esse descritor é obrigatório para todos os dispositivos que suportam a versão 2.0 (*high-speed*) e por conseguinte a versão 1.1 (*full-speed*). Esse descritor armazena a informação da velocidade que atualmente não está em uso e é alterado quando há uma mudança na versão USB em operação.
- **Descritor de Configuração:** esse descritor é obrigatório para qualquer dispositivo. O descritor de configuração contém informações sobre as funcionalidades do dispositivo como consumo de potência e número de interfaces suportadas. Apenas um descritor de configuração está ativo por vez.
- **Descritor de Configuração de Outra Velocidade:** esse descritor é obrigatório para todos os dispositivos que suportam ambas as versões 2.0 (*high-speed*) e 1.1 (*full-speed*). O descritor contém o mesmo tipo de informação que o descritor de configuração com a diferença de armazenar a configuração para a velocidade que não está sendo utilizada no momento.
- **Descritor de Interface:** esse descritor é obrigatório para qualquer dispositivo. O descritor contém informação sobre a quantidade de *endpoints* da interface, o identificador da interface e, a classe, subclasse e protocolo da interface. Um dispositivo pode ter várias interfaces ativas ao mesmo tempo ou mutuamente exclusivas.
- **Descritor de *Endpoint*:** esse descritor é opcional para qualquer dispositivo. Esse descritor possui o endereço do *Endpoint*, a direção do fluxo de dados, o tipo da transação utilizada, o tamanho máximo de cada pacote e, o intervalo de tempo de latência. O *Endpoint 0* não possui descritor porque é obrigatório, assume-se que trabalha com transações de controle e a informação necessária para ele é transmitida pelo descritor de dispositivo.
- **Descritor de *String*:** esse descritor é opcional para qualquer dispositivo. Ele contém informações textuais utilizadas para complementar as informações de cada um dos demais descritores.

Existem diversos outros descritores que são utilizados de acordo com as especificidades do dispositivo, aqui foram apresentados apenas os descritores padrão.

2.2 – A PONTE USBN9603

2.2.1 – Descrição Funcional

O USBN9603 é um circuito integrado fabricado pela National Semiconductor's que implementa uma ponte USB, ou seja, uma conexão elétrica entre o barramento USB e um barramento paralelo mais simplificado como o de um microcontrolador, por exemplo. A figura 2.1 mostra o diagrama de blocos do dispositivo.

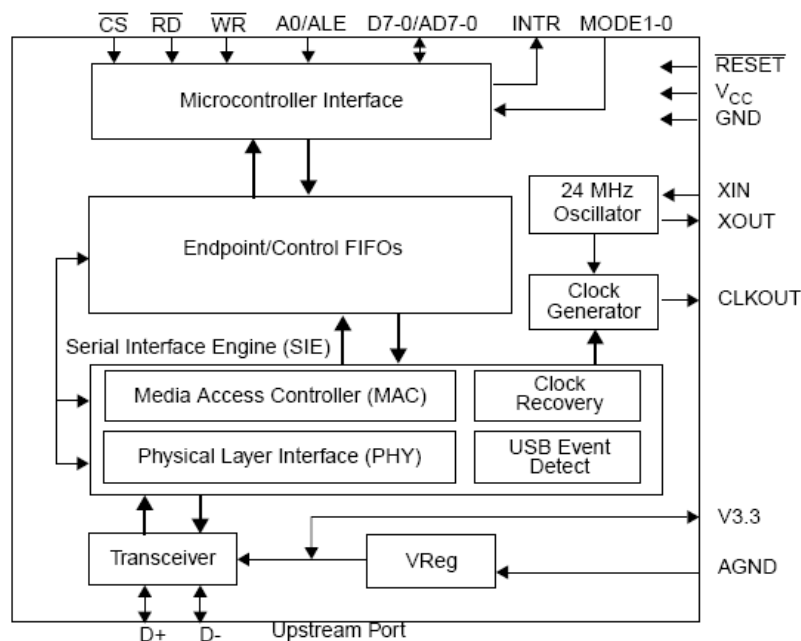


FIGURA 2.1: DIAGRAMA DE BLOCOS DO USBN9603 [12]

A arquitetura do USBN9603 é composta por uma interface serial para manipular as transmissões USB, um conjunto de *buffers* de *Endpoints* e uma série de registradores de *status* e de controle. O microcontrolador pode acessar os *buffers* de *Endpoints* e os registradores de status e controle a partir dos endereços de 00h até 3Fh via barramento externo local. O acesso ao barramento externo local pode ser acessado de três maneiras:

- Paralela multiplexada: essa maneira de acesso realiza a leitura ou a escrita de um *byte* em um ciclo de barramento. O endereço é armazenado com o sinal ALE e os dados com RD ou WR. A maioria dos microcontroladores com barramentos de dados externos pode utilizar esses sinais com pouca ou até mesmo nenhuma lógica adicional.

- Paralela não multiplexada: o acesso não multiplexado transfere ambos dados e endereço nos pinos D₇ a D₀ mas em ciclos de barramento separados. No primeiro ciclo de barramento é enviado o endereço e no segundo ciclo ocorre a transferência de dados do ou para o circuito integrado. Nesse tipo de acesso, para minimizar a quantidade de acessos ao barramento, o USBN9603 suporta um modo *burst* onde o microcontrolador escreve o endereço inicial para o dispositivo e então transmite ou recebe múltiplos *bytes* para endereços consecutivos. A interface paralela suporta também transferências de acesso direto à memória (DMA).
- Serial síncrona *Microwire*: esse modo de acesso é uma solução para microcontroladores que não possuem barramento de dados paralelo. *Microwire* é uma interface serial síncrona que usa quatro linhas: duas linhas de dados SIN (entrada serial) e SOUT (saída serial), CS (seleção) e SYNC (relógio). Comandos, endereços e bytes de dados trafegam bit a bit pelas linhas de dados usando as transições na linha SYNC como referência de tempo. O microcontrolador externo controla o relógio. Não há uma frequência mínima necessária e os sinais não necessitam sequer possuir a mesma frequência, apesar disso é recomendável que a frequência seja alta para manter o tráfego USB.

O USBN9603 é compatível com as versões 1.0 (*low-speed*) e 1.1 (*full-speed*) do padrão USB. Ele implementa, além do *Endpoint 0* com fila de oito *bytes*, três *Endpoints* de entrada (*Endpoint 1, 3 e 5*) com sessenta e quatro *bytes* cada e três *Endpoints* de saída (*Endpoint 2, 4 e 6*) também com sessenta e quatro *bytes* cada.

A qualquer tempo a ponte USBN9603 estará em um dos seguintes estados: *NodeOperational*, que é operação normal; *NodeSuspend*, que é a operação suspensa pela inatividade do barramento USB; *NodeResume*, que é o dispositivo retornando do modo *NodeSuspend*; e, *NodeReset*, que é a reinicialização do dispositivo. A transição de cada estado é disparada pelo *firmware*.

O uso do *Endpoint 0* se dá por um conjunto de registradores de controle e de status e uma fila, acessada por dois registradores TXD0 e RXD0, que é usada para as transações bidirecionais de controle. A fila do *Endpoint 0* é capaz de armazenar um único pacote de oito *bytes* que foi recebido ou que será transmitido. Um pacote escrito na fila é transmitido quando um *token IN* endereçado ao *Endpoint 0* é recebido. Se uma condição de erro é detectada o pacote permanece na fila para retransmissão no próximo *token IN*. O conteúdo da fila pode ser limpo para permitir a resposta a um *token OUT* ou para escrever um novo dado na fila para o

próximo *token IN*. Se um *token OUT* é recebido e não há a ocorrência de erros, o *firmware* é informado que a fila recebeu um dado.

Os *Endpoints 1, 3 e 5* são utilizados para transmissão do USBN9603 para o *host* e possuem registradores de controle, de *status* e de acesso à fila associada a cada um. A transmissão de dados inicia com um processo de *flush* que consiste em limpar a fila e informar ao dispositivo que um novo pacote será iniciado. Os *bytes* de dados são então colocados na fila através dos registradores TXD1, TXD3 e TXD5 para os *Endpoints 1, 3 e 5* respectivamente. Após o preenchimento dos dados do pacote é informado ao dispositivo o tipo de transação a ser utilizada e o tipo de pacote e o mesmo é então preparado e enviado ao *host*.

Os *Endpoints 2, 4 e 6* são utilizados para a recepção de dados enviados pelo *host* e possuem registradores de controle, de *status* e de acesso à fila associada a cada um. Quando o pacote de dados chega sem erros na ponte USBN9603 ela se encarrega de avisar ao microcontrolador que a fila de recepção, acessível pelos registradores RXD2, RXD4 e RXD6 para os *Endpoints 2, 4 e 6* respectivamente, contém dados. O processo de flush deve ser utilizado na recepção para limpar a fila e permitir ao dispositivo receber novos pacotes de dados.

A figura 2.2 ilustra a hierarquia dos registradores do dispositivo

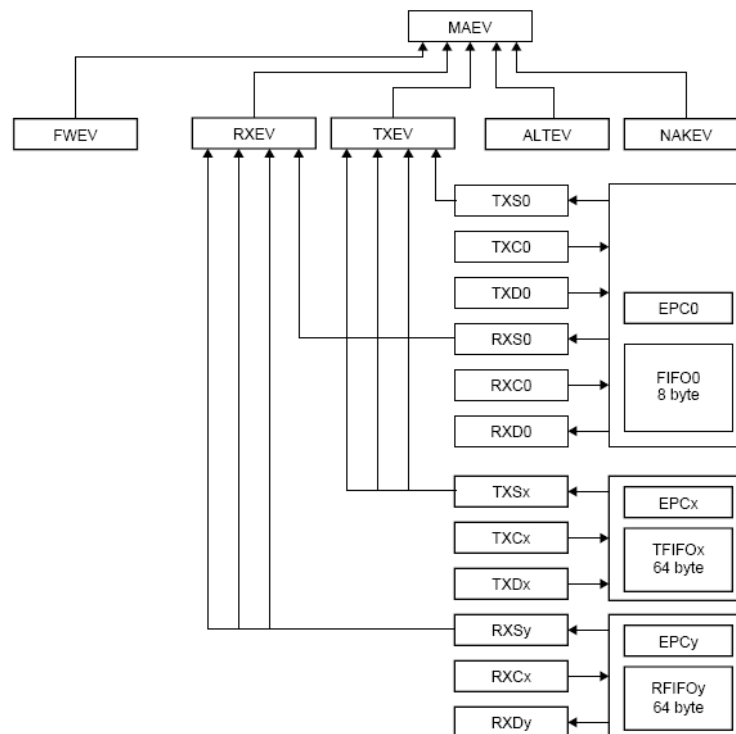


FIGURA 2.2: HIERARQUIA DOS REGISTRADORES DO USBN9603 [12]

Os registradores cujos nomes iniciam com ECP são responsáveis pelo controle dos *Endpoints*.

Os registradores cujos nomes iniciam com TXS, TXC e TXD são responsáveis por informar o *status* da transmissão, realizar o controle da transmissão e acessar a fila de transmissão respectivamente.

Os registradores cujos nomes iniciam com RXS, RXC e RXD são responsáveis por informar o *status* da recepção, realizar o controle da recepção e acessar a fila de recepção respectivamente.

Os registradores cujos nomes terminam com EV são responsáveis por avisar quais eventos ocorreram na comunicação USB.

2.2.2 – A Solução Adotada para a Transmissão dos Sinais ECG

A solução adotada para o USBN9603 na placa didática Pentacontroladora leva em consideração que o microcontrolador utilizado possui barramento paralelo de oito bits, dessa forma, a ponte é utilizada no modo paralelo não multiplexada. O barramento de dados paralelo da ponte USBN9603 é acessado pelo microcontrolador no endereço 85XXH. Para escrever na ponte o endereço de seus registradores o bit menos significativo do endereço 85XXH deve estar em nível alto e para acessar dados nos registradores da ponte o bit menos significativo do endereço deve estar em nível baixo. Uma combinação possível para acessar a ponte é primeiro enviar o endereço do registrador da ponte para 8501H e depois acessar o conteúdo do registrador a partir do endereço 8500H.

A placa didática Pentacontroladora disponibiliza dois *jumbers*. Um deles controla qual a versão USB (1.0 ou 1.1) será utilizada através da ligação do resistor de *pull-up* a uma das duas linhas de dados D+ ou D-. O outro controla se a alimentação da linha VBus é fornecida pelo barramento USB ou se é fornecida pela alimentação padrão da placa Pentacontroladora.

Um cristal de 24MHz foi utilizado juntamente com um circuito oscilador composto por dois capacitores e um resistor.

A saída INTR da ponte foi ligada a um flip-flop que registra as mudanças no seu estado indicando a ocorrência de um evento para que o microcontrolador possa tratá-lo.

CAPÍTULO 3 – ELETROCARDIOGRAMA (ECG)

3.1 – A FISIOLOGIA DO CORAÇÃO

O eletrocardiograma é o traçado da atividade elétrica do coração captada na superfície da pele. Os sinais elétricos são gerados pelo músculo cardíaco a partir da polarização e despolarização de suas células. Esse traçado é muito utilizado na medicina como uma importante ferramenta para a análise de cardiopatias e outros fatos relevantes à análise do músculo cardíaco.

Um eletrocardiograma normal é composto por cinco ondas que representam cada etapa da polarização e despolarização das cavidades cardíacas, os átrios e os ventrículos. As ondas detectadas são nomeadas em P, Q, R, S, T e U e subdivididas em onda P, complexo QRS, onda T e onda U. A figura 3.1 mostra um traçado típico de ECG

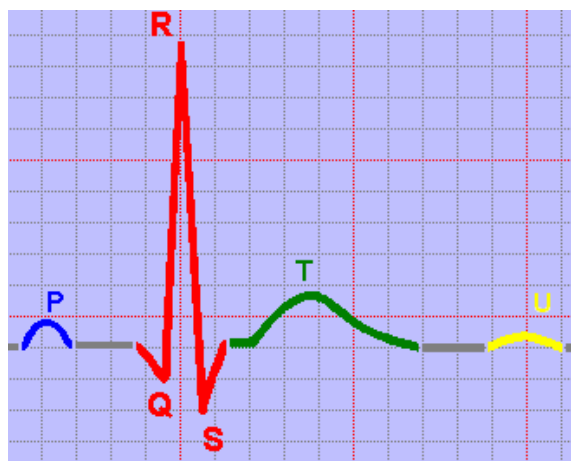


FIGURA 3.1: ONDAS P, Q, R, S, T E U EM UM TRAÇADO DE ECG

Estes sinais são descritos abaixo:

- A onda P representa o potencial elétrico responsável pela despolarização do átrio que ocorre pouco antes de sua contração.
- O complexo QRS representa o potencial que se propaga pelos ventrículos e que é responsável pela sua despolarização que também ocorre pouco antes da contração.
- A onda T representa o potencial de polarização dos ventrículos, que ocorre quando este está relaxado.
- A forma de onda U nem sempre é detectada em um sinal ECG e é gerada pelos potenciais tardios do movimento de dilatação do coração e suas artérias.

A disposição dos eletrodos no corpo permite a obtenção de diferentes amplitudes e formatos das ondas. Na medicina foram padronizadas posições de eletrodos que constituem um canal. Em um ECG padrão geralmente são obtidos doze canais, ou seja, doze diferentes visões do coração.

Em condições normais a duração média das ondas P, complexo QRS e ondas T não excedem a 0,11s, 0,10s e 0,15s respectivamente e têm amplitude máxima de 0,3mV, 3,0mV e 1mV respectivamente.

Para os canais que se utilizam de eletrodos nos dois braços e em uma das pernas a amplitude média da onda P é por volta de 0,1mV a 0,3mV, do complexo QRS é em torno de 1mV e da onda T entre 0,2mV e 0,3mV.

A frequência cardíaca de um ser humano varia de 30 a 220 batimentos por minuto o que equivale a 0,5 a 3,67Hz.

3.2 – PROJETO DO CIRCUITO ANALÓGICO

Nesse trabalho o projeto do circuito eletrônico para aquisição de sinais de ECG tem os seguintes requisitos: adquirir o sinal analógico de baixa amplitude, o amplificar e o filtrar permitindo assim a ligação ao conversor AD da placa Pentacontroladora.

O sistema deve estar apto a receber sinais de tensão entre 0,1mV e 3,0mV combinado com uma componente DC de $\pm 300\text{mV}$ – resultado do contato pele-eletrodo – e uma tensão de modo comum de 1,5V – resultado da diferença de potencial entre os eletrodos e o terra.

A banda de frequências útil de um sinal de ECG varia, de acordo com a aplicação, de 0,5 Hz a 50Hz para unidades de monitoramento em UTI e acima de 1kHz para medidas de potencial tardio (detecção de marca-passo). Um ECG clínico padrão tem largura de banda de 0,05Hz a 100Hz.

O sinal de ECG pode ser perturbado por diversos tipos de artefatos e fontes de perturbação. Os principais são:

- Interferência da rede elétrica: 60Hz e harmônicas de ordem superior.
- Contato dos eletrodos: a variação do contato do eletrodo com a pele causa um deslocamento da linha de base do sinal.
- Contração muscular: os sinais típicos de eletromiografia (EMG) são gerados e misturados aos sinais de ECG.
- Respiração: a respiração causa o deslocamento da linha de base do sinal.

- Interferência eletromagnética: outros dispositivos elétricos e eletrônicos podem causar interferência com os fios ligados aos eletrodos atuando como antenas.

Foi projetado um circuito para aquisição de 1 canal dos sinais ECG composto por cinco módulos de modo a minimizar os efeitos indesejáveis: um módulo amplificador diferencial, um módulo filtro passa-altas, um módulo filtro passa-baixas associado a um amplificador, um módulo para gerar a referência da perna direita e um módulo para gerar as tensões de alimentação e o terra comum para a alimentação dos demais módulos. O esquemático de ligação entre os módulos está mostrado na figura 3.2.

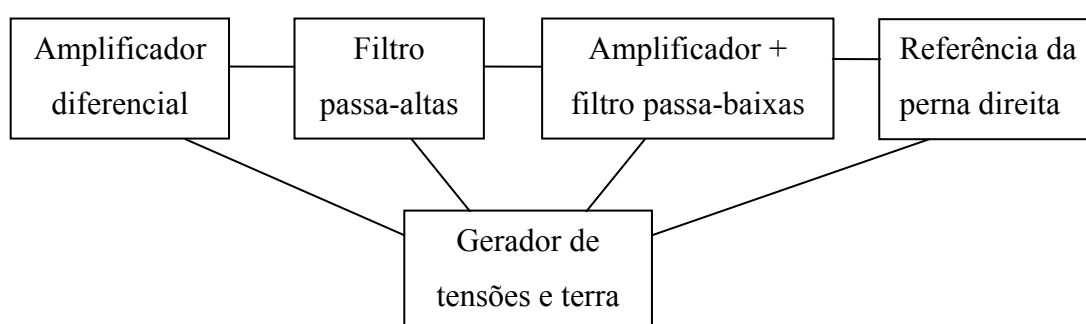


FIGURA 3.2: ESQUEMÁTICO DOS MÓDULOS PROJETADOS PARA O CIRCUITO DE ECG

O circuito foi projetado para obter um ganho de tensão de 1000V/V a partir dos sinais de entrada e para operar na faixa de frequências de 0,10Hz a 30Hz.

A figura 3.3 apresenta o circuito projetado para a aquisição dos sinais de ECG.

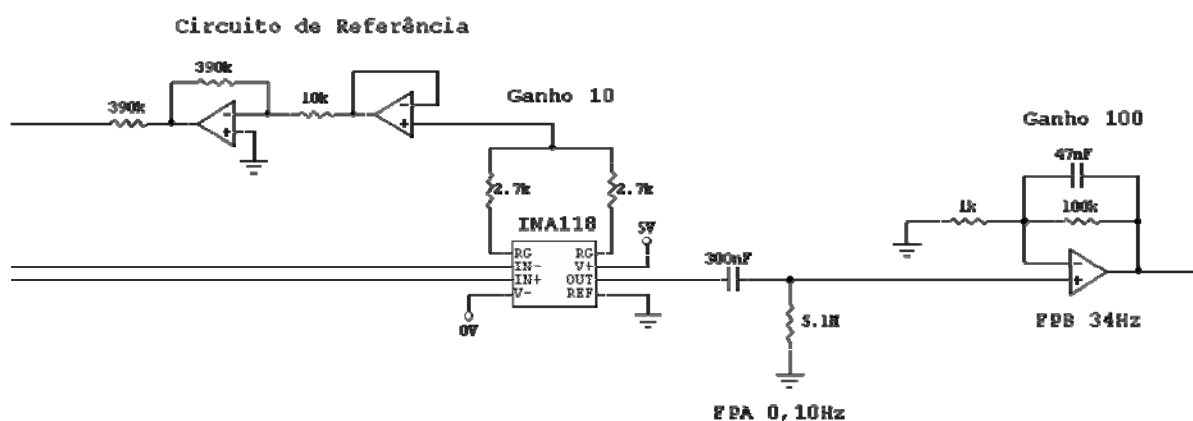


FIGURA 3.3: CIRCUITO PARA MEDIÇÃO E CONDICIONAMENTO DOS SINAIS DE ECG.

3.2.1 – Módulo Amplificador Diferencial

Para o amplificador diferencial foi escolhido o circuito integrado INA118 da Texas Instruments. Na verdade esse dispositivo é um amplificador de instrumentação com ganho regulável e que possui alta sensibilidade a sinais biomédicos, alta precisão, alto CMRR (taxa de rejeição de modo comum) e requer baixa potência para seu funcionamento.

A entrada desse módulo é composta pelos eletrodos que são colocados nos braços ou ombros da pessoa a ser examinada.

Segundo a folha de dados fornecida pela fabricante Texas Instruments o ganho do dispositivo é ajustado a partir de uma resistência colocada entre seus pinos 1 e 8. A equação utilizada para o cálculo do ganho é dada pela Eq 3.1.

$$G = 1 + \frac{50k\Omega}{R_G} \quad (\text{Eq 3.1})$$

Para esse módulo projetamos um ganho de 10V/V. A equação Eq 3.2 mostra o valor necessário do resistor:

$$10 = 1 + \frac{50 \times 10^3}{R_G} \therefore R_G = \frac{50 \times 10^3}{10 - 1} \therefore R_G = 5,56k\Omega \quad (\text{Eq 3.2})$$

Como o valor obtido para R_G não é um valor comercial e como necessitamos de uma tensão para ser utilizada no módulo de referência da perna direita cujo valor é a metade da tensão entre os terminais 1 e 8 do dispositivo foram escolhidos dois resistores no valor de 2,70k Ω cada. Esse valor foi escolhido porque é o valor comercial cuja montagem em série (5,40k Ω) mais se aproxima do valor de projeto obtido para R_G . Com esse valor temos o ganho real dado pela equação Eq 3.3.

$$G = 1 + \frac{50 \times 10^3}{5,40 \times 10^3} \Rightarrow G = 10,26V/V \quad (\text{Eq 3.3})$$

A figura 3.4 mostra o esquemático do circuito do módulo amplificador diferencial.

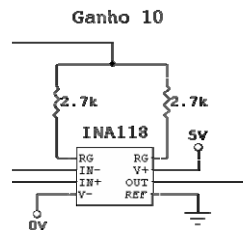


FIGURA 3.4: CIRCUITO MÓDULO AMPLIFICADOR DIFERENCIAL

3.2.2 – Módulo Filtro Passa-altas

A frequência de corte desejada para esse filtro é de 0,10Hz para remover as componentes DC geradas pelo contato dos eletrodos com a pele. Para tanto projetou-se um filtro RC de primeira ordem.

A entrada desse módulo filtro passa-altas é a saída do módulo amplificador diferencial.

A relação entre os valores de resistência, capacitância e frequência de corte é dada pela equação Eq 3.4.

$$\omega_c = \frac{1}{RC}, \omega_c = 2\pi f_c \therefore f_c = \frac{1}{2\pi RC} \text{ (Eq 3.4)}$$

Para esse projeto o valor da capacitância, por ser mais restrito no mercado de componentes, foi fixado. Partindo do valor comercial de capacitor disponível de 100nF e realizando uma associação em paralelo de três componentes e com a frequência de 0,10Hz temos o valor de resistor necessário dado pela equação Eq 3.5.

$$0,10 = \frac{1}{2.\pi.R.300.10^{-9}} \therefore R = \frac{1}{1,88.10^{-7}} = 5,30M\Omega \text{ (Eq 3.5)}$$

O valor comercial que mais se aproxima do valor projetado é de 5,10MΩ. Com esse valor de resistência a frequência de corte é pouco alterada como mostrado na equação Eq 3.6.

$$f_c = \frac{1}{2.\pi.5,10.10^6.300.10^{-9}} = 0,104Hz \text{ (Eq 3.6)}$$

A figura 3.5 mostra o esquemático do circuito do filtro passa-altas.

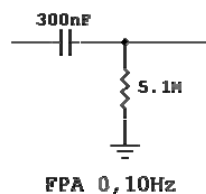


FIGURA 3.5: CIRCUITO MÓDULO FILTRO PASSA ALTAS

3.2.3 – Módulo Amplificador + Filtro Passa Baixas

O módulo amplificador diferencial possui um ganho de aproximadamente 10V/V. Como precisamos de um ganho total de 1000V/V para o circuito este módulo foi projetado para um ganho de 100V/V. Associado ao amplificador há um filtro passa-baixas RC de primeira ordem com frequência de corte de 30Hz. Este valor da frequência foi escolhido porque eliminará grande parte do ruído de 60Hz da rede elétrica injetado no sistema e não perderá grande quantidade de informação necessária nesse trabalho, ainda didático.

O amplificador operacional escolhido foi o amplificador com entradas a transistor de efeito de campo (FET) do modelo TL064 que possui alta impedância de entrada, baixo consumo de potência e que é fácil de ser encontrado no mercado.

A entrada desse módulo é a saída do módulo filtro passa-altas. A associação de resistores e capacitor ligados à entrada inversora do amplificador operacional formam o filtro passa baixas de primeira ordem conforme mostrado na figura 3.6.

Para o cálculo do ganho de 100V/V utilizamos um amplificador com realimentação negativa na configuração não-inversora e os dois resistores R_1 e R_2 . Fixamos o valor do resistor R_1 ligado à entrada negativa do amplificador e ao terra do circuito no valor de 1k Ω . Com base nesse valor utilizamos a equação Eq 3.7 para encontrar o valor do resistor R_2 que liga a entrada negativa com a saída do amplificador.

$$G = 1 + \frac{R_1}{R_2} \therefore 100 = 1 + \frac{R_1}{1 \cdot 10^3} \therefore R_1 = 99k\Omega \text{ (Eq 3.7)}$$

O valor comercial mais próximo do valor de projeto é de 100k Ω . Com esse valor de resistência temos ainda que calcular o capacitor que forma o filtro passa baixas RC de primeira ordem. A equação Eq 3.8 mostra esse cálculo e o valor da capacitância necessária.

$$f_c = \frac{1}{2 \cdot \pi \cdot R \cdot C} \therefore 30 = \frac{1}{2 \cdot \pi \cdot 100 \cdot 10^3 \cdot C} \therefore C = 53nF \text{ (Eq 3.8)}$$

O valor comercial de capacitor mais próximo do valor de projeto é 47nF. Com esse valor comercial temos então a real frequência de corte mostrada na equação Eq 3.9.

$$f_c = \frac{1}{2 \cdot \pi \cdot 100 \cdot 10^3 \cdot 47 \cdot 10^{-9}} = 33,86Hz \text{ (Eq 3.9)}$$

A figura 3.6 mostra o esquemático do circuito do módulo amplificador mais filtro passa-baixas.

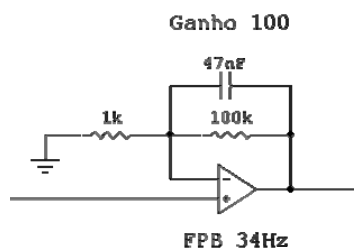


FIGURA 3.6: CIRCUITO MÓDULO AMPLIFICADOR + FILTRO PASSA BAIXAS

3.2.4 – Módulo Referência da Perna Direita

Para minimizar a interferência de 60Hz e melhorar a qualidade do sinal na saída do sistema foi adotada a utilização do circuito de referência da perna direita. Esse circuito consiste de uma realimentação entre paciente e eletrodo onde poucos microamperes são conduzidos pelo corpo humano do eletrodo na perna direita até os eletrodos nos braços. Essa realimentação coloca em níveis baixos o ruído de modo comum através do aumento do CMRR (razão de rejeição de modo comum) resultante.

Um inconveniente dessa técnica é a possibilidade de oscilação caso ocorra uma mudança de fase no sinal que percorre o corpo humano. Esse caso indesejado ocorre pela presença de altas frequências de oscilação no sinal ECG. No caso desse trabalho esse inconveniente não deve ocorrer uma vez que o sinal está filtrado para a banda de frequências de 0,10Hz a 33Hz.

Nesse módulo a entrada positiva de um primeiro estágio amplificador é conectada à metade da tensão entre os terminais que determinam o ganho no módulo amplificador diferencial. No caso desse trabalho o sinal foi ligado no ponto de junção dos dois resistores de 2,7k Ω do módulo amplificador diferencial.

O primeiro estágio é um circuito seguidor de tensão, ou seja, um buffer. A saída desse buffer é ligada a um segundo estágio amplificador na configuração inversora com ganho de 39V/V e com a corrente de saída limitada por um resistor de 390k Ω .

O dispositivo escolhido para esse módulo foi o amplificador OPA2277 da Texas Instruments que possui altíssima precisão, baixo consumo de potência e considerável largura de banda.

A saída desse módulo é ligada ao eletrodo colocado na perna direita ou na parte inferior direita da barriga da pessoa a ser examinada.

A figura 3.7 mostra o esquemático do circuito de referência da perna direita.

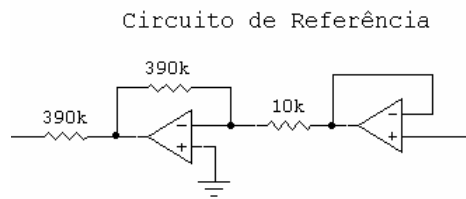


FIGURA 3.7: CIRCUITO MÓDULO DE REFERÊNCIA DA PERNA DIREITA

3.2.5 – Módulo de Gerador de Tensões de Alimentação e Terra

Nesse trabalho a pretensão foi de aproveitar a alimentação estabilizada de 5V da placa Pentacontroladora. Essa característica obrigou o projeto de um circuito para alimentar os amplificadores dos demais módulos e para gerar um terra virtual do circuito. Para permitir uma tensão negativa e outra positiva a partir de +5V e 0V projetou-se um circuito retificador a partir de um diodo Zener de 2,4V, um resistor de 180Ω ligado em série com o diodo de modo a polarizá-lo e dois capacitores, um cerâmico de 100nF e um eletrolítico de $10\mu\text{F}$, ligados em paralelo com o diodo a fim de minimizar o ruído na tensão que servirá de terra virtual.

A junção entre o diodo e o capacitor é a tensão estabilizada que foi designada o terra virtual do circuito. A tensão original 0V foi ligada como alimentação negativa para os demais módulos amplificadores uma vez que ela tem uma diferença de potencial de -2,4V em relação ao terra virtual. A tensão original de +5V foi ligada como alimentação positiva para os demais módulos amplificadores já que ela apresenta uma diferença de potencial de +2,6V em relação ao terra virtual.

A Figura 3.8 apresenta o circuito de terra virtual e alimentação positiva e negativa.

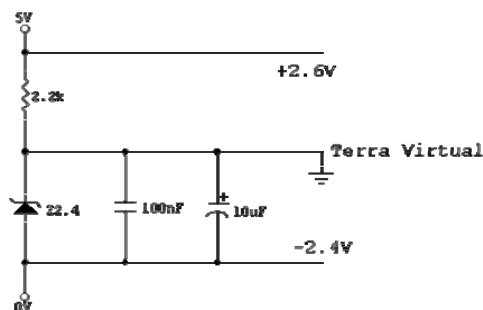


FIGURA 3.8: CIRCUITO MÓDULO GERADOR DE TENSÕES DE ALIMENTAÇÃO E TERRA

CAPÍTULO 4 – CIRCUITOS ADICIONAIS

4.1 – O CONVERSOR ADC12048

O conversor A/D escolhido seguiu a opção adotada para a placa Pentacontroladora (P5C), o ADC12048 da fabricante National, uma vez que, além de já ter sido projetado para uso na P5C, é uma opção de baixo custo, boa resolução, e relativa disponibilidade no mercado.

4.1.1 – Descrição Funcional

O ADC12048 é um conversor A/D com resolução de doze *bits* mais um *bit* de sinal, dotado de barramento paralelo de treze ou oito *bits*, programável internamente, e que atinge uma taxa de amostragem de até 216kHz, em modo síncrono ou assíncrono. Ainda, possui circuito *Sample-and-Hold* (S/H) interno, e multiplexador analógico de oito canais, permitindo inclusive que seja digitalizada a diferença entre dois sinais. A figura 4.1 apresenta o diagrama de blocos do conversor.

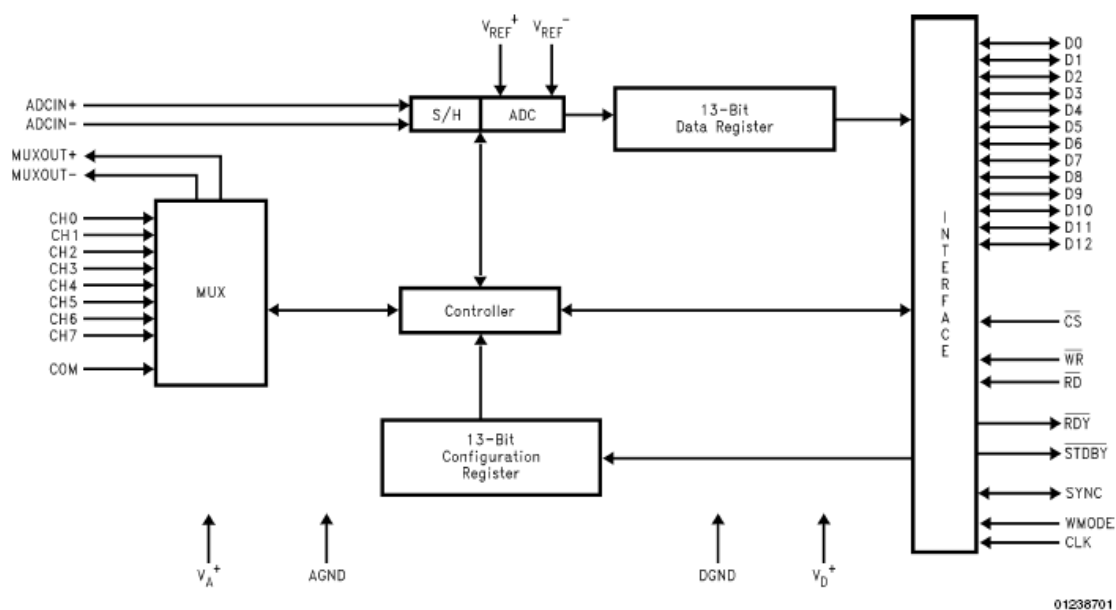


FIGURA 4.1: DIAGRAMA DE BLOCOS DO ADC12048 [11]

O conversor possui dois registradores de treze *bits*, um para somente-escrita, e outro para somente-leitura. O primeiro é o registrador de configuração, que armazena toda a configuração do dispositivo, onde cada *bit* tem uma função específica, como mostra a tabela 4.1.

MSB				LSB								
b ₁₂	b ₁₁	b ₁₀	b ₉	b ₈	b ₇	b ₆	b ₅	b ₄	b ₃	b ₂	b ₁	b ₀
BW	COMMAND FIELD			SYNC	HB	SE	ACQ TIME		MUX ADDRESS			

TABELA 4.1: DESCRIÇÃO DOS *BITS* DO REGISTRADOR DE CONFIGURAÇÃO DO ADC12048 [11]

O *bit* BW define se o conversor AD irá trabalhar com barramento de treze bits (BW em nível alto) ou oito bits (BW em nível baixo).

Os *bits* de b₁₁ a b₉ (*command field*), indicam o comando que será executado pelo conversor AD, colocando-o em um de seus estados possíveis. O conversor ADC12048 possui cinco comandos que são apresentados na tabela 4.2.

b ₁₁ b ₁₀ b ₉	Comando
000	<i>Standby</i>
001	<i>Ful-Cal</i>
010	<i>Auto-zero</i>
011	<i>Reset</i>
100	<i>Start</i>

TABELA 4.2: COMANDOS DISPONÍVEIS AO CONVERSOR ADC12048 [11]

Cada comando possui as seguintes atribuições:

- *Standby*: este comando coloca o ADC12048 em um modo de operação de baixo consumo de potência (75μW a 5V). Quando neste modo o pino #STDBY é colocado em nível baixo, indicando o estado *Standby*. Qualquer outro comando tira o conversor AD desse estado, e o pino #STDBY é colocado imediatamente em nível alto. O pino #RDY é colocado em baixo logo que o conversor estiver pronto para operação. Ao ser iniciado, por padrão, o ADC12048 permanece neste modo de operação.

- *Ful-Cal*: este comando calibra o conversor de forma a reduzir erros de *offset* e linearidade. Nesta operação é realizada a medida dos erros de *offset* e de linearidade por oito vezes para o cálculo dos coeficientes de correção.
- *Auto-zero*: este comando faz com que o ADC12048 realize apenas uma leitura do erro de *offset* e utilize este valor para a correção do erro.
- *Reset*: este comando força o conversor AD a entrar no estado de pronto para operação, colocando o pino #RDY em nível baixo. Este comando pode ser utilizado para interromper qualquer operação que o conversor esteja executando ou para retirá-lo do modo *standby*.
- *Start*: este comando inicia o processo de conversão.

O bit SYNC determina se o dispositivo trabalhará em modo síncrono (SYNC em nível alto) ou assíncrono (SYNC em nível baixo). Quando o conversor opera no modo síncrono o pino SYNC funciona como entrada, controlando o tempo de aquisição e o início da conversão. Quando o conversor opera no modo assíncrono o pino SYNC funciona como saída, informando o estado do conversor.

O bit HB só tem funcionalidade quando o conversor AD está operando no modo paralelo de oito bits. Quando colocado em nível alto indica ao conversor que a próxima escrita será efetuada no *byte* superior do registrador de comando.

O bit SE (*single-ended*), quando configurado em nível alto, faz com que o conversor retorne somente valores positivos. Quando na entrada é detectado um valor negativo o resultado da conversão é o valor zero.

Os bits de *acq time* definem, quando no modo assíncrono, a quantidade de batidas do relógio em que a aquisição se dará. Esses bits devem ser configurados de modo a haver tempo suficiente para carregar o capacitor do circuito S/H.

Já os bits designados por *mux address* (b_3 a b_0), determinam quais entradas do multiplexador analógico serão conectadas às entradas do conversor.

O segundo registrador é o registrador de dados. Nele é armazenado o resultado da conversão efetuada como mostrado na tabela 4.3. Toda leitura no ADC12048 retornará o conteúdo deste registrador. Quando configurado para operar em um barramento de dezesseis bits, a leitura no conversor AD retornará nos bits de 12 a 0 os bits deste registrador, e quando configurado para operar em barramento de 8 bits, retornará na primeira leitura o *byte* menos significativo (b_7 a b_0), e na segunda leitura o restante dos bits (b_{12} a b_8).

MSB					LSB							
b ₁₂	b ₁₁	b ₁₀	b ₉	b ₈	b ₇	b ₆	B ₅	b ₄	b ₃	b ₂	b ₁	b ₀
sign	Conversion Data											

TABELA 4.3: DESCRIÇÃO DOS *BITS* DO REGISTRADOR DE DADOS DO ADC12048 [11]

O processo de conversão ocorre de maneira diferente para os modos síncrono e assíncrono:

- **Modo Assíncrono (*sync-out*):** nesse modo deve-se escrever no registrador de configuração os canais do multiplexador analógico desejados, o valor correspondente ao tempo de aquisição desejado, nível baixo no *bit* SYNC, e os *bits* correspondentes ao comando *Start*. Após a completa leitura do registrador de dados, o ADC12048 inicia a conversão, respeitando o tempo de aquisição programado. No início da aquisição o pino #RDY é colocado em nível alto, seguido pelo pino SYNC no início da conversão. Os dois pinos, #RDY e SYNC, são levados a nível baixo no fim da conversão, indicando que uma nova informação pode ser colocada no registrador de configuração, se desejado, e que um novo valor pode ser retornado através da leitura de seu registrador de dados. Novamente, após a leitura completa do registrador de dados, inicia-se novo ciclo de conversão.
- **Modo Síncrono (*sync-in*):** nesse modo deve-se escrever no registrador de configuração, os canais analógicos desejados e colocar o bit SYNC em nível alto. Neste modo, após a borda de subida do pino SYNC ou a completa leitura do registrador de dados, o pino #RDY é colocado em nível alto. Após este ponto, o pino SYNC controla o tempo de aquisição e o início de cada conversão. A aquisição se dá quando o pino SYNC está em nível baixo. Quando esse pino vai para nível alto o período de aquisição termina e inicia-se o processo de conversão, o pino #RDY é então colocado em nível alto. Ao término da conversão #RDY é colocado em nível baixo, indicando seu fim. Neste ponto pode-se escrever novo comando no registrador de dados, se necessário. Um novo período de aquisição se inicia após a completa leitura do registrador de dados.

4.1.2 – A Solução Adotada para a Aquisição dos Sinais ECG

A solução adotada para o ADC12048 na placa didática Pentacontroladora leva em consideração que o microcontrolador utilizado possui barramento de oito bits, dessa forma, o conversor é utilizado no modo paralelo de oito bits, permitindo a compatibilidade entre seus barramentos. O barramento de dados do ADC12048 é acessado pelo microcontrolador no endereço 8402H.

Para a entrada CLK, utilizou-se o sinal XTAL da placa Pentacontroladora, dividido por dois por meio de um flip-flop, ou seja, o próprio relógio do microcontrolador dividido por dois.

O sinal #RDY é o sinal responsável por gerar a interrupção no microcontrolador. O sinal passa por um flip-flop que registra o evento para que ele possa ser recuperado pelo microcontrolador. O sinal #STDBY não pode ser recuperado pelo microcontrolador e está disponível apenas como ponto de teste.

Através de um *jumper* pode-se conectar o pino SYNC do conversor AD a um endereço de memória externa, acessível ao microcontrolador, ou a um circuito externo. Essa configuração permite que o ADC12048 seja utilizado no modo síncrono com o sinal SYNC controlado pelo microcontrolador, ou controlado por um circuito externo, ou mesmo em modo assíncrono. Também há um *jumper* que possibilita utilizar as tensões 0V e 5V de alimentação da placa como tensões de referência ou utilizar tensões externas.

O circuito de apoio ao ADC12048 conta ainda com um circuito de aquisição com um pré-amplificador e um amplificador de ganho variável. Este circuito leva um sinal analógico que chega por um conector compatível com conector de microfones comuns, ao conversor AD no seu canal 7. O circuito pré-amplificador insere um nível DC de 2,4V no sinal, de forma a permitir que o ADC12048 trabalhe na faixa de -2,5V a 2,5V.

4.2 – O MOSTRADOR GRÁFICO DE CRISTAL LÍQUIDO (GLCD) AGM-12864K

4.2.1 – Descrição Funcional

O mostrador gráfico nada mais é que um dispositivo composto por uma matriz de pontos em um visor de cristal líquido associada a um controlador e uma memória. O

dispositivo possui um barramento de vinte terminais utilizados para receber os sinais lógicos e as tensões de alimentação.

O GLCD AGM-12864K é composto por duas matrizes de pontos simétricas de 64 x 64 dispostas lado a lado formando um mostrador de 128 x 64 pontos com luz de fundo. As figuras 4.2 e 4.3 apresentam o diagrama de blocos do circuito de alimentação e do circuito lógico respectivamente.

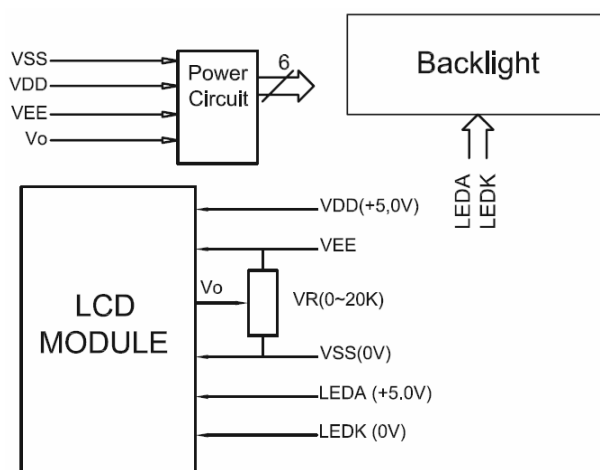


FIGURA 4.2: DIAGRAMA DE BLOCOS DA ALIMENTAÇÃO DO GLCD AGM-12864K [9]

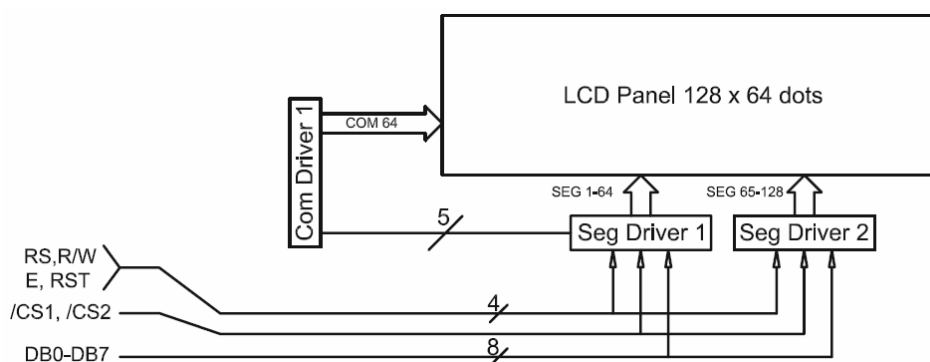


FIGURA 4.3: DIAGRAMA DE BLOCOS DO CIRCUITO LÓGICO DO GLCD AGM-12864K [9]

A alimentação do dispositivo é composta pelos sinais VEE, Vo, VDD, VSS, LEDA e LEDK. Os sinais VEE, e Vo são utilizados para a regulação de contraste dos pontos no mostrador. Os sinais VDD e VSS são as tensões de alimentação e referência do dispositivo respectivamente. Por fim, os sinais LEDA (anodo dos LEDs) e LEDK (catodo dos LEDs) são utilizados para a regulação da iluminação de fundo do visor.

A parte lógica do dispositivo é composta pelos sinais RS, R/W, E, RST, /CS1, /CS2 e os sinais de DB0 a DB7 que compõem o barramento de dados de oito *bits*. As combinações

dos sinais RS e R/W indicam ao dispositivo se o conteúdo do barramento são comandos ou dados a serem lidos do visor ou escritos no mesmo. O sinal E indica ao dispositivo que este deve interagir com o barramento. O sinal RST reinicia o microcontrolador interno do dispositivo. Os sinais /CS1 e /CS2 quando em nível baixo indicam ao dispositivo qual das matrizes de 64 x 64 pontos deve ser acessada.

Cada matriz de 64 x 64 pontos é dividida em oito páginas de 64 x 8 pontos. Cada uma das sessenta e quatro colunas de uma página corresponde a um *byte* onde cada *bit* corresponde a um ponto da matriz. O *bit* em nível alto indica o ponto acesso e o *bit* em nível baixo indica o ponto apagado. Os pontos de cima para baixo são representados pelos *bits* menos significativos para os mais significativos. A figura 4.4 ilustra o arranjo da matriz.

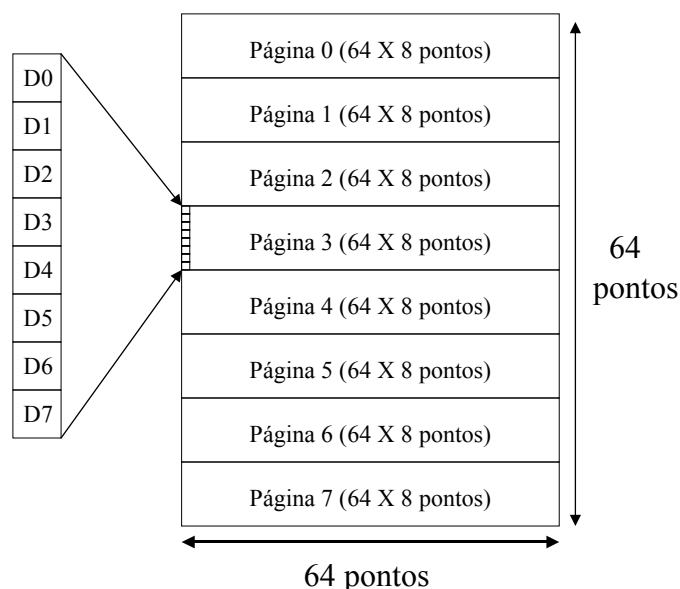


FIGURA 4.4: COMPOSIÇÃO DA MATRIZ 64 X 64 DO GLCD AGM-12864K [7]

As operações que podem ser realizadas com o dispositivo, como já descrito, são indicadas pela combinação dos sinais RS e R/W. Essas operações estão mostradas na tabela 4.4.

RS	R/W	Operação
0	0	Envio de instruções para o GLCD
0	1	Leitura do byte de estado
1	0	Escrita de um dado
1	1	Leitura de um dado

TABELA 4.4: OPERAÇÕES DISPONÍVEIS NO MOSTRADOR GRÁFICO AGM-12864K

O AGM-12864K pode executar quatro instruções. A tabela 4.5 apresenta as instruções disponíveis e os *bytes* que representam essas instruções.

Instrução	D7	D6	D5	D4	D3	D2	D1	D0
Ligar / desligar mostrador	0	0	1	1	1	1	1	L
Definir linha início	1	1	A5	A4	A3	A2	A1	A0
Definir endereço X (da página)	1	0	1	1	1	X2	X1	X0
Definir endereço Y	0	1	Y5	Y4	Y3	Y2	Y1	Y0

TABELA 4.5: INSTRUÇÕES DISPONÍVEIS NO MOSTRADOR GRÁFICO AGM-12864K [7]

- Ligar / desligar mostrador: essa instrução utiliza o *bit* L em nível alto para ligar o mostrador e o *bit* L em nível baixo para desligá-lo. Essa instrução não altera o conteúdo das matrizes de pontos.
- Definir linha de início: essa instrução utiliza os *bits* de A5 a A0 como endereço da linha do topo do mostrador. Na verdade essa configuração permite girar horizontalmente o conteúdo dos pontos das matrizes.
- Definir endereço X (da página): essa instrução utiliza os bits de X2 a X0 para definir a página que será utilizada para as operações de leitura e de escrita de dados. Depois dessa instrução todas as leituras e escritas serão realizadas na página especificada até que seja novamente enviada uma instrução para modificar a página.
- Definir endereço Y: essa instrução utiliza os bits de Y5 a Y0 para definir uma das sessenta e quatro colunas de cada matriz de pontos. Após a execução dessa instrução é obrigatória a realização de uma operação de leitura de um dado que deve ser ignorado. A cada operação de leitura ou escrita de dados o valor da coluna é incrementado.

A operação de leitura do *byte* de estado coloca no barramento de dados um conjunto de *bits* que representam o estado atual do GLCD. O *bit* 7 em nível alto indica que o GLCD está ocupado e que não deve ser acessado com outras operações e em nível baixo indica que está pronto para o uso. O *bit* 5 em nível alto indica que visor está ligado e em nível baixo que o visor está desligado. O *bit* 4 em nível alto indica que o GLCD está sendo inicializado e em nível baixo indica que já foi inicializado.

A operação de escrita de um dado coloca o *byte* recebido do barramento de dados no visor respeitando a página configurada e a coluna representada pelo endereço Y anteriormente definido. A posição da coluna (endereço Y) é incrementada na realização dessa operação.

A operação de leitura de um dado coloca o *byte* que representa a coluna (endereço Y) e a página previamente definidas no barramento de dados. Essa operação também realiza um incremento no endereço Y.

4.2.2 – A Solução Adotada para Plotar os Sinais ECG

A solução adotada na placa “Pentacontroladora” para o mostrador gráfico de cristal líquido dispõe de um conjunto de oito *latches* e oito *buffers tri-state* para ligar o barramento de dados aos pinos DB7 a DB0 do mostrador com uma lógica que permita ler dados a partir dos *buffers tri-state* e escrever a partir dos *latches* através do endereço externo 8200H.

A solução implementa também um conjunto de *latches* e *buffers tri-states* para permitir ler e escrever os sinais de controle. O endereço externo compartilhado 8100H permite acessar através dos bits zero, um, dois, três, quatro e cinco os sinais E, RS, R/W, /CS1, /CS2 e RST respectivamente.

Através de dois *jumpers* e um potenciômetro pode-se definir o tipo do circuito de referência para o contraste dos pontos no visor. Isso permite o uso de variados modelos de mostradores quer eles gerem a tensão de referência negativa internamente quer eles necessitem de uma tensão negativa externa necessária para o contraste dos pontos.

Há também um potenciômetro para definir a tensão na entrada LEDA (anodo do LED) que é utilizada para regular o brilho da luz de fundo do visor.

4.3 – A MEMÓRIA EXTERNA 62256

4.3.1 – Descrição Funcional

O circuito integrado 62256 é uma memória estática desenvolvida em tecnologia CMOS organizada em 32768 palavras de oito bits. O diagrama de blocos da memória é mostrado na figura 4.5.

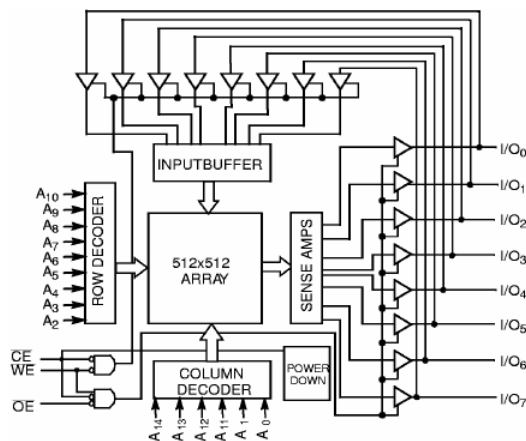


FIGURA 4.5: DIAGRAMA DE BLOCOS DA MEMÓRIA 62256 [10]

Os sinais de entrada de A14 a A0 são ligados ao barramento de endereços e representam os quinze bits necessários para endereçar 32768 palavras.

Os sinais de I/O7 a I/O0 são ligados ao barramento de dados e são utilizados para escrever na memória ou ler desta de acordo com o sinal /WE.

O sinal /CE indica para a memória que ela está ativa e deve realizar a escrita ou a leitura do dado constante no endereço. Ao colocar esse sinal em nível baixo a memória automaticamente entra em um modo de baixo consumo de energia.

O sinal /OE indica para a memória que o dado endereçado pelos pinos de A14 a A0 deve ser disponibilizado nos pinos de I/O7 a I/O0.

O tempo de acesso da memória 62256 é de 70ns.

4.3.2 – A Solução Adotada para Armazenar os Sinais ECG

A placa didática “Pentacontroladora” permite o uso de uma memória estática para armazenamento de dados a partir dos quinze *bits* menos significativos do barramento de endereços que são ligados aos pinos de A14 a A0. O barramento de dados está ligado aos pinos de I/O7 a I/O0. O sinal /CE é ligado ao *bit* mais significativo do barramento de endereços. Essa ligação permite a escrita e a leitura de dados dos a partir dos endereços externos de 0000H até 7FFFH.

A placa possui na verdade um módulo de memória EPROM e dois módulos de memória estática. Existem quatro *jumper*s para configurar o uso dessas memórias. Um *jumper* é utilizado para indicar se a EPROM possui um programa de *boot* ou não. Os outros três

jumpers indicam o tamanho de cada memória, se a EPROM é de 16kB ou de 32kB e se as SRAM são de 8kB ou de 32kB.

Nesse projeto a placa didática foi configurada para operar com a primeira EPROM contendo programa de *boot* e a segunda EPROM armazenando os sinais de ECG resultantes da digitalização.

CAPÍTULO 5 – PROGRAMAS

5.1 – FIRMWARE

5.1.1 – Laço Principal

O *firmware* utilizado na placa didática “Pentacontroladora” foi concebido de modo a permitir a enumeração USB na classe HID do sistema operacional Windows e após esse processo permitir a recepção de comandos e o envio de dados a partir do barramento USB, a aquisição e conversão dos sinais a partir do conversor AD, a armazenagem na memória RAM estática reservada como memória de dados e a plotagem dos sinais armazenados no mostrador gráfico de cristal líquido.

Após realizar a inicialização dos registradores, da ponte USBN9603, do Conversor AD ADC12048 e das interrupções o programa entra no laço principal.

Ao entrar no laço principal a ponte USBN9603 já está no estado operacional e as interrupções que ocorrem de imediato realizam a enumeração da placa didática junto ao sistema operacional Windows no PC.

Foi definido que o PC comandaria a placa “Pentacontroladora” a partir do envio de caracteres no padrão ASCII via barramento USB. A tabela 5.1 descreve esses comandos.

Comando	Descrição
“D”	Inicializa o processo de digitalização
“P”	Para o processo de digitalização
“L”	Inicia o processo de plotagem no mostrador gráfico
“U”	Inicia o processo de transmissão dos dados armazenados na memória da placa para o PC via USB

TABELA 5.1: COMANDOS UTILIZADOS NA COMUNICAÇÃO PC – “PENTACONTROLADORA”

No laço principal, o microcontrolador executa os comandos de decisão que aguardam a chegada de um comando via USB. Ele permanece executando essa verificação até que o comando “D” chegue e seja detectado. Após a detecção do comando o processo de

Após a parada do processo de digitalização o programa aguarda a chegada de um comando via USB e alterna na comparação entre os comandos “L”, “U” e “D” para realizar a plotagem no GLCD, o envio via USB ou iniciar nova digitalização respectivamente. A figura 5.1 apresenta o fluxograma básico do laço principal.



Chegou

Comando di installazione via USB?

Digitalização e Armazenagem na Memória

Não

O sinal *sync-ad* é gerado pelo microcontrolador que utiliza o contador-temporizador 0 de forma a gerar interrupções em uma frequência de 3600Hz (aproximadamente a cada 277 μ s) , e uma posição de memória é utilizada como contador para que se possa dividir esta frequência por um valor entre 1 e 256. A rotina de interrupção provocada por este contador é responsável por inverter o sinal *sync-ad*, assim, quando esta função é habilitada é gerada uma onda quadrada com frequência igual à metade da frequência de ocorrência da mesma, podendo então, nesta configuração, variar de 7Hz a 1800Hz, o que é suficiente para os propósitos deste projeto. É pertinente ressaltar, no entanto, que a placa “Pentacontroladora” tem capacidade para frequências muito superiores a esta, podendo chegar até a 216kHz, a frequência máxima que o ADC12048 consegue digitalizar.

Na borda de descida do sinal *sync-ad*, é iniciado o processo de digitalização pelo conversor AD, que ao terminar, leva o sinal #RDY para nível baixo, gerando uma interrupção externa no microcontrolador. A partir desta interrupção compartilhada o microcontrolador lê o resultado da conversão em duas etapas, primeiro o LSB do dado convertido, e em seguida o MSB, uma vez que o AD trabalha com precisão de doze *bits* (um *byte* e meio) em um barramento de dados de oito *bits*. Após as duas leituras no conversor AD o sinal #RDY vai automaticamente para o nível alto indicando que o conversor está pronto para receber comandos ou iniciar nova conversão e os dois *bytes* resultados da conversão são então movidos para a memória RAM externa, para serem tratados posteriormente. Então, no tempo determinado pelo sinal *sync-ad*, inicia-se uma nova conversão. Este processo segue até que a memória esteja cheia, ou que o comando para parar a digitalização seja percebido.

Para que a parte do *firmware* referente ao conversor AD seja integrada à parte referente à comunicação USB e à comunicação visual do estado do hardware através dos LEDs, são necessárias algumas medidas. Primeiramente, como a interrupção externa é compartilhada entre o ADC12048 e o USBN9603, é necessária a identificação do dispositivo que solicitou a interrupção, através da leitura do *buffer tri-state* que espelha o conteúdo dos *flip-flops* responsáveis por registrar e armazenar os pedidos, no tempo oportuno, e então é necessário o desvio para a rotina de tratamento de interrupção específica para atender a cada um dos dispositivos, que deverá se encarregar de apagar o respectivo pedido.

A comunicação visual do estado do *hardware*, utiliza como símbolos o piscar de diferentes LEDs, onde cada LED indica uma situação, conforme mostrado na tabela 5.2. Para a geração de um sinal que possa controlar a frequência com que os LEDs piscam, foi utilizado o contador-temporizador 0, o mesmo utilizado para gerar o sinal *sync-ad*. Como há a pretensão da frequência de oscilação de *sync-ad* ser diferente da frequência dos LEDs, a

solução empregada foi a de utilizar diferentes registradores para a geração individual das duas bases de frequência de modo a manter a frequência com que a interrupção é chamada constante.

Cor do LED	Significado
Verde (VD)	Pronto para digitalizar
Amarelo (AM)	(Sem função)
Vermelho (VM)	Enviando dados via USB ou no LCD
Azul (AZ)	Dados armazenados na memória
Branco (BR)	Em funcionamento
Bicolor – Verde (BVD) (estático)	Digitalizando
Bicolor – Vermelho (BVM) (estático)	Memória cheia

TABELA 5.2: DESCRIÇÃO DOS ESTADOS INFORMADOS PELOS LEDS

5.1.3 – Enumeração, Transmissão e Recepção USB

A parte do *firmware* referente à interface USB foi baseada no trabalho realizado por SALOMÃO & FRIEDMAN [4], que propõe uma interface USB para a placa “Pentacontroladora”, em uma versão mais antiga, utilizando o USBN9603, e não será detalhado aqui, mas apenas as alterações efetuadas com o intuito de compatibilizá-lo às necessidades deste projeto. A figura 5.2 apresenta o fluxograma básico deste *firmware*.

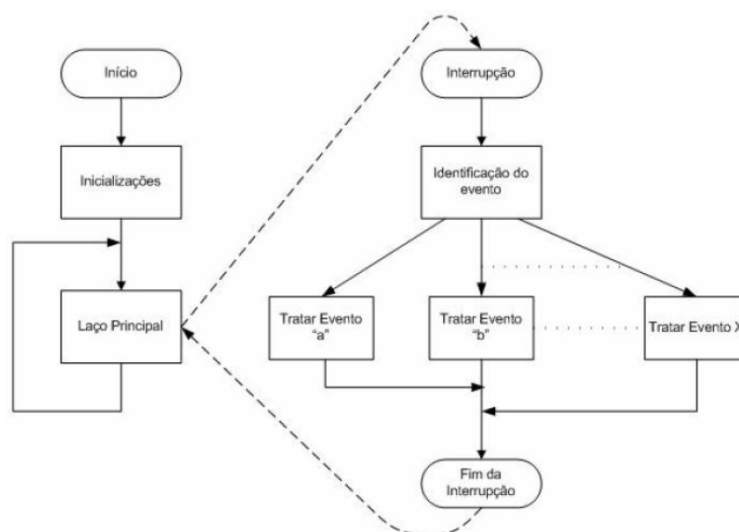


FIGURA 5.2: FLUXOGRAMA BÁSICO PARA PARTE USB DO *FIRMWARE* [4]

O processo de inicialização e enumeração da interface USB permaneceu inalterado, a não ser pela definição de um novo tamanho de pacote, visto que o tamanho que havia sido definido originalmente de 10 *bytes*, era ineficiente para a aplicação a que o *hardware* se destinava, que exigiria um alto volume de transmissão de dados, no sentido da placa “Pentacontroladora” para o PC, no momento da transferência dos dados digitalizados. Foi então definido, através das constantes de inicialização do USB no descritor do HID, o tamanho de 64 *bytes* por pacote.

A definição de um pacote com 64 *bytes* criou um inconveniente devido à simetria de tamanho entre os pacotes de transmissão e recepção. Isto porque o pacote de recepção também é definido com o tamanho de 64 *bytes*, dos quais, somente o primeiro *byte* é efetivamente utilizado como comando enviado pelo PC que preenche automaticamente os 63 *bytes* restantes com o valor 00H. O maior inconveniente, no entanto, não é o PC perder tempo preenchendo um pacote de transmissão com valores inúteis, já que estes são equipados com microprocessadores muito rápidos se comparado ao microcontrolador 8051, mas sim, este perder um tempo precioso de seu processamento na leitura e identificação de 63 *bytes* que deverão ser descartados. Assim, a solução adotada foi de alterar a rotina de interrupção que trata o evento de recepção USB de forma que esta recebesse o primeiro *byte* da ponte USBN9603, e descartasse todo o resto, além de realizar a operação de *flush* no *Endpoint* de recepção de forma que este eliminasse toda sua fila de recepção. Com este procedimento, conseguimos um ganho de rendimento pelo controlador, evitando a perda de tempo com um processamento inútil.

O controle do tamanho do pacote de dados enviado da placa didática para o PC deve ser tratado na placa. Apesar de pacotes de qualquer tamanho entre 1 e 64 *bytes* puderem ser enviados pela ponte USBN9603, o envio de pacotes com tamanho diferente do informado no descritor HID coloca os dados no barramento USB mas os mesmos não são percebidos pelo sistema operacional Windows como um pacote válido.

O *firmware* programa a ponte USBN9603 para trabalhar com transações USB do tipo *Interrupt* na versão 1.1 (*Full-Speed*). Os pacotes de dados transmitidos são alternados entre os tipos *data0* e *data1* de modo a detectar possíveis erros de transmissão como solicitado pelo protocolo USB. Além do *Endpoint 0* de controle, apenas dois *Endpoints* da ponte são habilitados a partir do envio dos descritores para o PC: o *Endpoint 1*, de transmissão e o *Endpoint 2*, de recepção. Os eventos de transmissão e recepção que incluem a chegada de um

pacote ACK bem como o envio desse tipo de pacote são tratados na rotina chamada pela interrupção externa responsável por eventos na ponte USB.

5.1.4 – Plotagem no GLCD

A parte do *firmware* que trabalha com o mostrador gráfico de cristal líquido implementa diversas rotinas para possibilitar a plotagem dos sinais adquiridos pelo conversor AD.

Como a resolução do mostrador é de 128 pontos na horizontal por 64 pontos na vertical os valores obtidos pelo conversor AD que variam de 0 a 4095 devem ser mapeados em uma escala de 0 a 63. Para realizar esse mapeamento foi realizado um deslocamento de 6 bits à direita no número obtido do conversor AD (12 bits) que equivale a uma divisão por 64 transformando o valor em um número de 6 bits, ou seja, a faixa de valores desejada de 0 a 63.

Foram implementadas rotinas para ligar e desligar o mostrador que consistem de comandos enviados ao microcontrolador do GLCD.

Ao inicializar o mostrador o mesmo apresenta-se com todos os pontos preenchidos. Para possibilitar a visualização dos dados escritos foi implementada uma rotina para limpar todo o mostrador. Essa rotina apaga todos os pontos a partir da varredura de todas as páginas e colunas do mostrador.

Para permitir a plotagem contínua do sinal sem que haja sobreposição foi implementada uma rotina para limpar uma coluna do mostrador desde o ponto 0 até o ponto 63. Essa rotina é chamada antes da plotagem do valor atual sempre na coluna imediatamente posterior à coluna onde o valor é plotado.

Uma rotina para escrever apenas um ponto no mostrador foi implementada para facilitar o trabalho da plotagem de linhas no mostrador. Por sua vez, para a realização dessa rotina foi necessária a implementação de duas sub-rotinas uma para definir a coluna (valor Y) e outra para definir a página (valor X). Depois da definição de coordenadas é necessário fazer uma leitura da coluna na página a fim de manter o estado dos demais pontos, aplicar uma máscara que acenda o ponto desejado e só então o valor do byte com o estado total da coluna é enviado ao mostrador.

A rotina para plotagem de linhas utiliza o princípio da união de pontos. Dois registradores armazenam a posição do ponto atual. A rotina recebe dois parâmetros que correspondem à coordenada de destino da linha. Para verificar a quantidade de incrementos

em cada coordenada para cada iteração foi necessário realizar uma divisão de 16 *bits* em um microcontrolador de 8 *bits* o que foi solucionado coma a implementação de uma rotina de divisão utilizando subtrações sucessivas. De posse do valor do resto da divisão foi necessário definir qual quadrante do sistema de coordenadas a linha estará presente e depois dessas duas definições o incremento / decremento de cada coordenada do ponto foi definido. A rotina realiza as iterações necessárias até completar a linha e no final atualiza o valor dos registradores que armazenam a coordenada do ponto atual.

A rotina para plotagem da linha dependendo das coordenadas pode ser demasiadamente demorada e por isso surgiu a necessidade de criar uma rotina para verificar o registrador de estado do microcontrolador do GLCD e verificar se o mesmo pode receber comandos.

5.2 – APLICATIVO WINDOWS

O aplicativo Windows foi desenvolvido na linguagem de programação Microsoft Visual Basic utilizando-se de funções da API do Windows e da Classe HID desenvolvidas em C++.

O aplicativo tem a responsabilidade de procurar a placa “Pentacontroladora” dentre os dispositivos USB que utilizam a classe HID. Depois de encontrá-la tem a função de detectar as características da placa enviadas por ela própria quando do processo de enumeração. Ainda depois desses processos a conexão com a placa finalmente é aberta e o envio de comandos é permitido ao usuário.

O envio de comando simplesmente envia um caracter no formato ASCII para a placa. A função da API da classe HID é responsável por preencher o pacote com o tamanho solicitado pelo descritor HID enviado ao PC durante o processo de enumeração.

Quando o comando de recepção de dados é enviado e os dados chegam da placa para o PC os mesmos são armazenados em memória em um laço até que o sinalizador de que a transmissão terminou seja detectado. Após a recepção dos dados os mesmos são plotados na tela do computador.

O sinalizador de término de transmissão foi definido como dois *bytes* consecutivos contendo o valor 0FFH. Esse sinalizador foi definido levando em consideração que valores do conversor AD que possuem apenas 12 *bits* jamais poderiam atingir 16 *bits*, todos com valor 1.

O laço da rotina de recepção é interrompido por um objeto do tipo Evento disponibilizado pela API do Windows. Esse tipo de objeto tem um sinalizador que associado á função de recepção do USB é ativado quando há a percepção de que um pacote válido chegou ao sistema operacional. Quando ocorre a sinalização do evento um pacote na sua totalidade (64 *bytes*) é lido e é realizada a comparação de dois *bytes* 0FFH como já descrito. No caso da não ocorrência dos dois bytes de sinalização de término o sistema entra novamente no laço para esperar a chegada de um novo pacote.

A figura 5.3 mostra a tela do programa aplicativo.

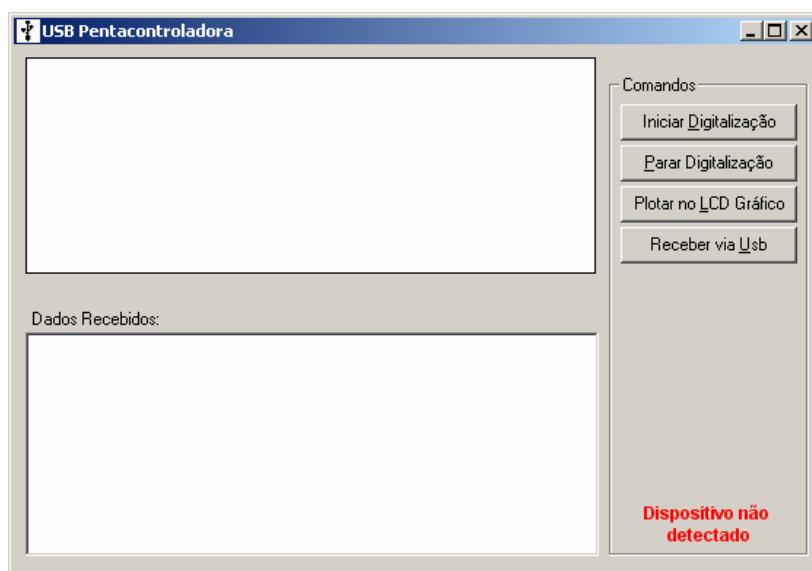


FIGURA 5.3: TELA DO PROGRAMA APLICATIVO

CAPÍTULO 6 – RESULTADOS

6.1 – ELETROCARDIOGRAMA (ECG)

A montagem do circuito de ECG foi realizada em placa de contatos o que compromete um pouco a aquisição devido à magnitude dos sinais de entrada. Apesar disso o resultado obtido foi bastante satisfatório. O sinal de ECG foi capturado e medido no osciloscópio no seu formato padrão com as ondas P, complexo QRS e onda T bem definidas. A figura 6.1 mostra o sinal adquirido no osciloscópio.

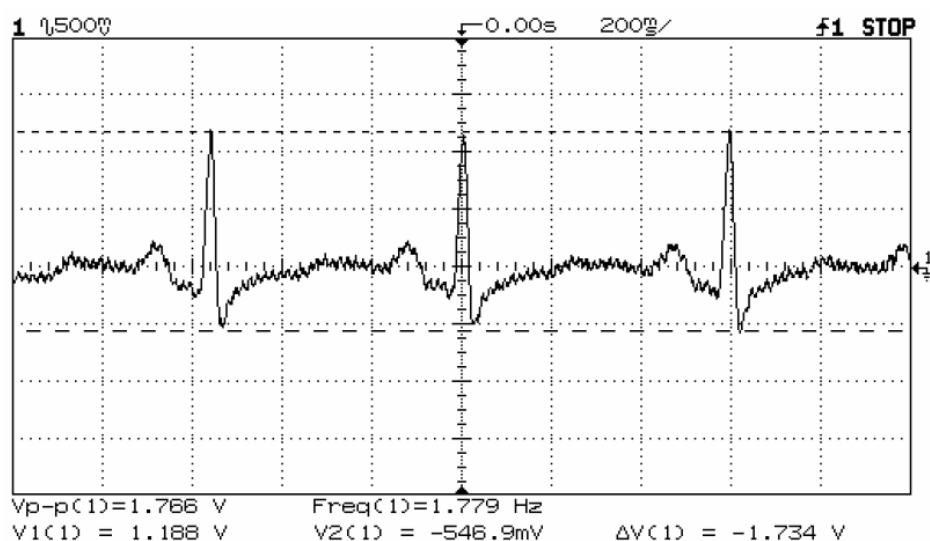


FIGURA 6.1: SINAL DE ECG OBTIDO NO OSCILOSCÓPIO

Na figura 6.1 notamos a presença de sinais de alta frequência e baixa amplitude no sinal captado. Isso mostra que o filtro passa-baixas implementado não foi suficiente para tornar o sinal de ECG mais puro. Em uma análise do módulo amplificador mais filtro passa-baixas mostrado na figura 3.6 percebe-se que o capacitor se comportará como um curto-circuito para frequências muito maiores que a frequência de corte do filtro e nesse caso o módulo aplicará ao sinal de alta frequência um ganho unitário. Um ajuste na montagem do circuito com a implementação de um filtro passa baixas de maior eficiência e atenuação pode resolver o problema.

O sinal visibilizado na figura 6.1 mostra a amplitude de aproximadamente 1,8V de pico-a-pico e a frequência de 1,779Hz equivalente a 107 batimentos cardíacos por minuto.

Esses dois parâmetros, amplitude e frequência, e principalmente suas variações são de fundamental importância para a avaliação médica do paciente e o sinal adquirido possibilitou a leitura dessas medidas.

Ao ampliar a janela de tempo de aquisição do sinal podemos perceber de maneira mais imediata a variação na linha de base do sinal devido principalmente ao efeito da respiração (movimentos do tórax) sobre o sinal de ECG. A figura 6.2 mostra essa variação que no caso do sinal mostrado foi menor que 500mV.

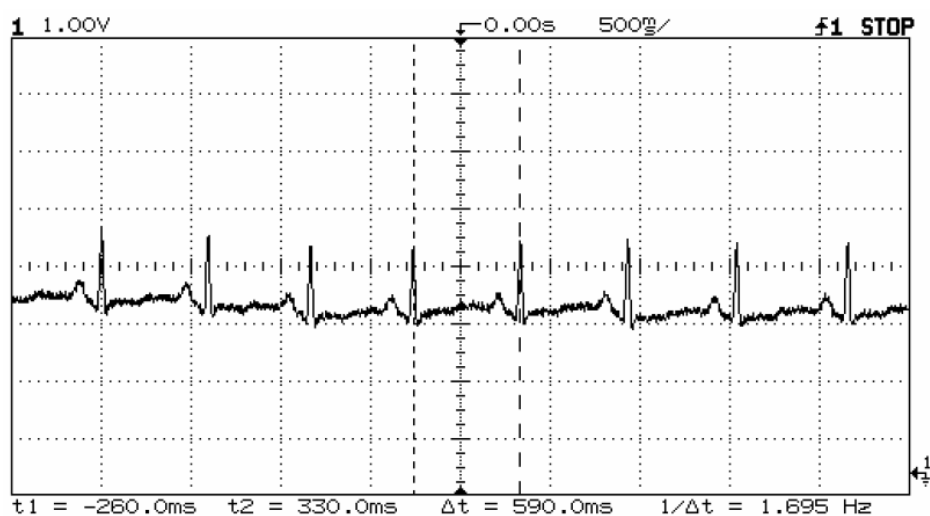


FIGURA 6.2: SINAL DE ECG OBTIDO NO OSCILOSCÓPIO COM JANELA DE TEMPO MAIOR

6.2 – DIGITALIZAÇÃO DOS SINAIS

A conversão analógico-digital dos sinais apresentou alguns comportamentos anômalos. Foram realizados diversos ensaios com sinais diferentes bem como com conversores ADC12048 diferentes e os resultados obtidos foram os mesmos: a digitalização gerou *bytes* que em parte correspondem ao sinal aplicado à entrada, mas que em outros instantes apresentou bytes que correspondem a um comportamento aparentemente aleatório.

A figura 6.3 mostra o resultado da conversão de uma senóide simétrica aplicada à entrada do dispositivo conversor AD. A figura 6.4 apresenta o resultado da digitalização de uma senóide assimétrica. A figura 6.5 apresenta o resultado da conversão de um sinal triangular. A figura 6.6 apresenta o resultado da conversão de uma onda quadrada. E por fim, a figura 6.7 apresenta o resultado da conversão do sinal de ECG.



FIGURA 6.3: RESULTADO DA DIGITALIZAÇÃO DE UM SINAL SENOIDAL SIMÉTRICO

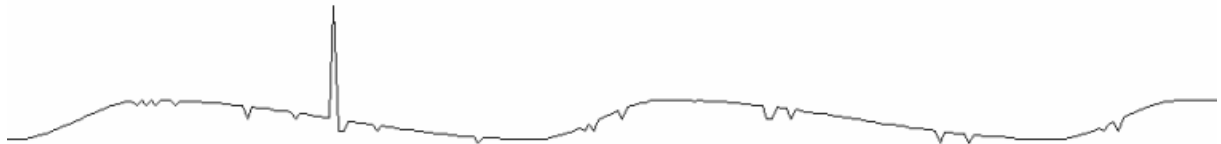


FIGURA 6.4: RESULTADO DA DIGITALIZAÇÃO DE UM SINAL SENOIDAL ASSIMÉTRICO

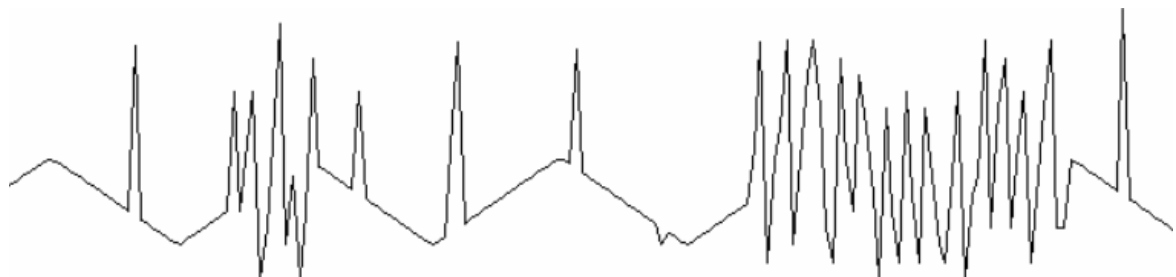


FIGURA 6.5: RESULTADO DA DIGITALIZAÇÃO DE UM SINAL TRIANGULAR

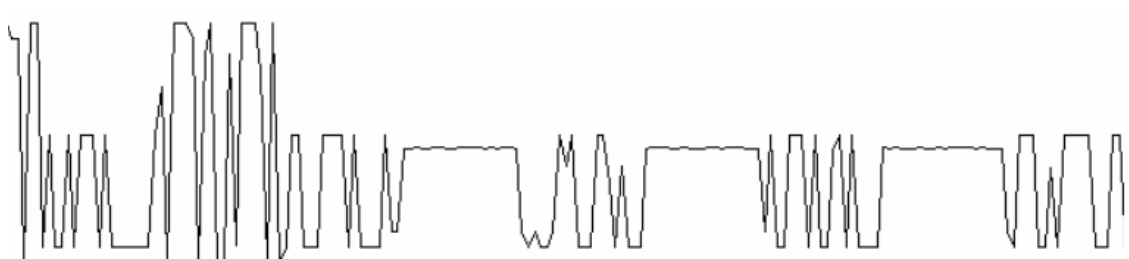


FIGURA 6.6: RESULTADO DA DIGITALIZAÇÃO DE UMA ONDA QUADRADA

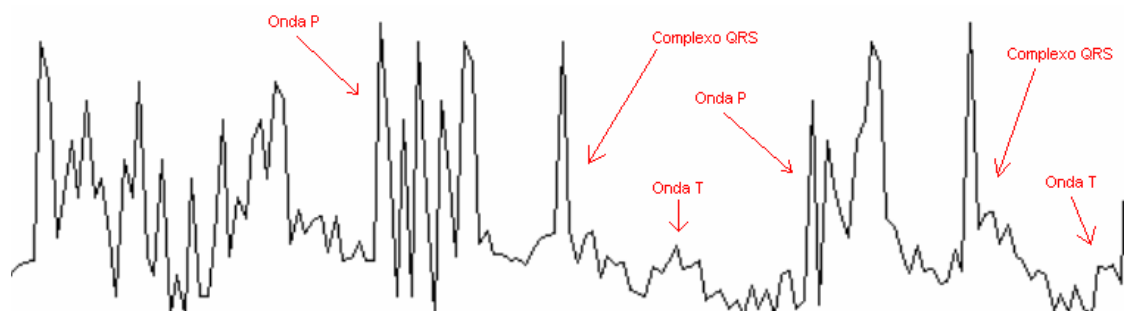


FIGURA 6.7: RESULTADO DA DIGITALIZAÇÃO DO SINAL DE ECG

A figura 6.7 mostra o resultado da digitalização do sinal de ECG. Apesar das distorções aparentemente aleatórias verificamos os padrões de onda esperados conforme destacado em dois segmentos de onda presentes na figura.

A característica aleatória das distorções observadas indica que pode haver algum problema com o *hardware*, no caso a montagem da placa didática uma vez que diversos componentes ADC12048 foram testados, ou, com o *firmware* onde o controle das interrupções do USB compartilhadas com o controle das interrupções do conversor AD pode fazer com que leituras no AD sejam realizadas sem que este esteja pronto para ser lido.

Ambas possibilidades foram checadas e testadas e não foi possível solucionar o problema.

Em trabalhos futuros uma análise minuciosa das soldas nos componentes placa didática e testes com pequenos trechos do código do *firmware* serão necessários para permitir a detecção da origem do problema e apontar possíveis soluções.

CAPÍTULO 7 – CONCLUSÃO

O circuito para amplificação e filtragem de um canal dos sinais de eletrocardiograma (ECG) foi projetado e implementado para operar aplicando um ganho de 1000V/V no sinal de entrada, filtrar o sinal numa banda de 0,1Hz a 33Hz aproximadamente. O resultado final do circuito foi satisfatório levando em consideração a ausência de cabos especiais com blindagem e a montagem realizada em placa de matriz de contatos.

O conversor A/D ADC12048 presente na placa didática “Pentacontroladora” foi programado para funcionar no modo síncrono com o microcontrolador MCS8051 também presente na placa informando ao conversor o momento da digitalização. O conversor não funcionou corretamente apresentando a conversão de sinais de aparência aleatória. A interrupção compartilhada da placa “Pentacontroladora” aparentemente funcionou corretamente informando ao microcontrolador quando a conversão foi concluída e não interferindo na interrupção solicitada pela ponte USB ao microcontrolador.

A plotagem dos sinais no mostrador gráfico de cristal líquido também foi bem sucedida. Inicialmente apenas pontos foram plotados devido à complexidade de cálculos exigidos para uma rotina que plotasse linhas. Algumas semanas depois um algoritmo foi implementado para a plotagem das linhas e o sinal ECG enfim pode ser observado em sua forma tradicional.

O envio dos dados adquiridos para um PC através do barramento USB também foi muito bem sucedido. O *firmware* utilizado como base da comunicação USB necessitou apenas de algumas poucas alterações para que a placa “Pentacontroladora” fosse enumerada porém essas alterações exigiram um estudo mais aprofundado da especificação do barramento, da especificação da ponte USBN9603 e o papel de cada um de seus registradores, de ferramentas de varredura do barramento e da classe HID do sistema operacional Windows. Após a enumeração da placa o desafio foi a transmissão e a recepção de dados no PC. O estudo de funções já definidas e os parâmetros necessários para seu correto funcionamento também se mostrou necessário. O resultado final foi muito bom, com o sistema aplicativo no PC enviando comandos para a placa “Pentacontroladora” que os interpretava corretamente e esta enviando os dados da conversão para o PC que recebia todos sem perda de pacotes.

Trabalhos futuros podem aproveitar o poder do protocolo USB para implementação de um eletrocardiógrafo digital que adquira e converta os sinais cardíacos em tempo real e em mais de um canal. Microcontroladores mais poderosos que o MCS8051 com maior frequência

de operação, e que implementem inclusive a comunicação USB e o conversor AD possibilitam a miniaturização do circuito e a aquisição de diversos canais simultaneamente. O baixo consumo desses microcontroladores permitiria ainda um sistema alimentado pelo próprio barramento USB. Essa solução seria muito útil para médicos realizarem exames de Eletrocardiograma em locais remotos ou de difícil acesso necessitando levar apenas um microcomputador *Laptop* que possua uma porta USB e o dispositivo.

Para que o circuito de ECG seja mais seguro para o paciente será necessário, em trabalhos futuros, projetar e implementar circuito um isolador óptico para os eletrodos ligados à pessoa para que esses façam parte de um circuito independente dos amplificadores, filtros, conversor AD e microcontrolador.

REFERÊNCIAS BIBLIOGRÁFICAS

- [1] AXELSON, Jan. *USB Complete*. 2. ed., Estados Unidos da América: Lakeview Research, 2001.
- [2] FEITOSA, Eumann Magalhães; FREITAS, Patrícia de Oliveira; FEITOSA, Virgínia Fernandes. *Proposta de Componentes de Sistema de Telemetria para Monitoramento de ECG e de EMG em Atividade Física*. Projeto Final de Graduação. Brasília: Universidade de Brasília, 2005.
- [3] HYDE, John. *USB Design by Example. A Practical Guide to Building I/O Devices*. Estados Unidos da América: Intel University Press, 2001.
- [4] SALOMÃO, Andrei & FRIEDMAN, Alberto de Carvalho. *Desenvolvimento de uma Interface USB para Comunicação com o Microcontrolador 8051*. Projeto Final de Graduação. Brasília: Universidade de Brasília, 2005.
- [5] SEDRA, Adel S.; SMITH, Kenneth C. *Microeletrônica*. 4. ed., São Paulo: Makron Books, 2000.
- [6] SMITH, Steven W. *The Scientist and Engineer's Guide to Digital Signal Processing*. 2. ed., Estados Unidos da América: California Technical Publishing. 1999.
- [7] ZELENOVSKY, Ricardo; MENDONÇA, Alexandre. *Microcontroladores: Programação e Projeto com a Família 8051*. 1. ed., Rio de Janeiro: MZ editora, 2005.
- [8] _____. *PC: Guia Prático de Hardware e Interfaceamento*. 3. ed., Rio de Janeiro: MZ editora, 2002.
- [9] AGT TECHNOLOGIES. *AGM-12864K SERIES*. S. Rev., Santa Rita do Sapucaí, 2004.

- [10] CYPRESS SEMICONDUCTOR CORPORATION. *CY62256 32Kx8 Static RAM*. Rev. 3, Estados Unidos da América, 1997
- [11] NATIONAL SEMICONDUCTORS. *ADC12048 12-Bit Plus Sign 216kHz 8-Channel Sampling Analog-to-Digital Converter*. S. rev., Estados Unidos da América: 2002.
- [12] NATIONAL SEMICONDUCTORS. *USB9603/USB9604 Universal Serial Bus Full Speed Node Controller with Enhanced DMA Support*. Rev. 1.3, Estados Unidos da América: 2003.
- [13] TEXAS INSTRUMENTS. *HANDBOOK OF OPERATIONAL AMPLIFIER APPLICATIONS*. S. rev., Estados Unidos da América: 2001.
- [14] TEXAS INSTRUMENTS. *INA118 Precision, Low Power INSTRUMENTATION AMPLIFIER*. S. rev., Estados Unidos da América: 1998.
- [15] TEXAS INSTRUMENTS. *OPA277 OPA2277 OPA4277 High Precision OPERATIONAL AMPLIFIERS*. S. rev., Estados Unidos da América: 1998.
- [16] TEXAS INSTRUMENTS. *TL061, TL061A, TL061B, TL062, TL062A, TL062B, TL064, TL064A, TL064B LOW-POWER JFET-INPUT OPERATIONAL AMPLIFIERS*. Rev. 2, Estados Unidos da América: 2004.