UNIVERSIDADE FEDERAL DE ITAJUBÁ
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Helton Hugo de Carvalho Júnior

Sistema embarcado de diagnóstico de eletrocardiograma utilizando *fuzzy clustering* e correlação

Tese submetida ao Programa de Pós-Graduação em Engenharia Elétrica como parte dos requisitos para obtenção do Título de Doutor em Ciências em Engenharia Elétrica.

Área de Concentração: Microeletrônica

Orientadores: Dr. Robson Luiz Moreno
Dr. Tales Cleber Pimenta

Dezembro de 2011
Itajubá – MG

Dedico este trabalho aos meus pais, Helton e Marly, e à minha esposa, Débora Bizinoto.

A diferença entre o possível e o impossível está na vontade humana.

Louis Pasteur

Agradecimentos

Agradeço a Deus por tornar possível o desenvolvimento deste trabalho, cercando-me de pessoas maravilhosas sem as quais eu nada seria.

Aos orientadores, Prof. Robson Luiz Moreno e Prof. Tales C. Pimenta, pela orientação no desenvolvimento deste trabalho e constante disposição em auxiliar.

Ao professor Evaldo Cintra Renó e aos colegas do Grupo de Microeletrônica da UNIFEI pelas valiosas contribuições dadas a esse trabalho;

A UNIFEI, em especial ao Grupo de Microeletrônica, pelo empréstimo da placa de desenvolvimento FPGA XILINX SPARTAN 3A, sem a qual este trabalho não seria possível.

Aos meus familiares e amigos, pelo apoio incondicional e compreensão pelos momentos de ausência enquanto desenvolvia este trabalho.

À CAPES, pelo apoio financeiro através do programa Demanda Social, ao CNPq e a Fapemig.

Meus mais sinceros agradecimentos.

Resumo

Este trabalho visa demonstrar a viabilidade e o desenvolvimento de um sistema embarcado de identificação de doenças cardíacas. O propósito do sistema é a aquisição de sinais cardíacos, originados de um eletrocardiograma conectado a um paciente, e a apresentação de um provável diagnóstico. Para isso algumas técnicas de processamento de sinais são utilizadas. O sistema recebe os sinais de derivações de um eletrocardiograma, transmitido por sensores posicionados em pontos específicos no corpo do paciente. Estes sinais são filtrados e processados de forma que na saída, deste sistema, seja exibido o possível diagnóstico do paciente em análise.

Para o funcionamento deste sistema será apresentada uma alternativa, visando melhorar seu funcionamento, que é a utilização do processo de *Fuzzy Clustering*, numa tradução direta: amostragem nebulosa, entretanto como esta nomenclatura não está consolidada o nome original será utilizado. Este processo de *Fuzzy Clustering* permite extrair as principais características de um sinal de entrada e, através destas características, fornecer regras que podem ser utilizadas em vários tipos de aplicações de controle e tomadas de decisões. Este trabalho demonstra a possibilidade de utilização deste processo para gerar os pontos que representam as características principais de um sinal de eletrocardiograma.

Para comprovar a funcionalidade do sistema, foi utilizado o método de correlação para comparar e obter o grau de similaridade entre o sinal em análise e os sinais de um banco de dados com o diagnóstico médico conhecido. A comparação que obtiver maior grau de similaridade, será reconhecida como o possível diagnóstico para o paciente.

O sistema foi implantado em uma placa XILINX Spartan®-3A Starter FPGA. Sendo a FPGA (Field-programmable gate array) configurada com o Microblaze® Soft-Core

Processor - XILINX Embeded Processor. Foram feitos testes reais deste sistema, utilizando uma base de dados pública, e os resultados foram comparados com figuras de méritos de outros trabalhos que utilizam outras técnicas para obter o diagnóstico de um paciente. Também foram verificados outros fatores, da implantação em hardware dedicado, que tornam este sistema mais eficiente utilizando as técnicas empregadas.

Abstract

This paper demonstrates the viability and development of an embedded system for identifying heart disease. The purpose of the system is the acquisition of cardiac signals, originating from an ECG connected to a patient, and the presentation of a likely diagnosis. To this some signal processing techniques are used. The system receives signals from an electrocardiogram derivations, transmitted by sensors placed at specific points in the patient's body. These signals are filtered and processed so that the output of this system, to show the possible diagnosis of the patient in analysis.

To operate this system will be presented an alternative to improve its functioning, which is the use of the process of Fuzzy Clustering. This process of Fuzzy Clustering allows to extract the main features of an input signal and, through these features, provide rules that can be used in a variety of control applications and decision making. This work demonstrates the possibility to use this process to generate the points that represent the main features of an ECG signal.

To prove the functionality of the system, we used the correlation method to compare and get the degree of similarity between the signal being analyzed and the signs of a database with known medical diagnosis. The comparison gets greater degree of similarity to be recognized as a possible diagnosis for the patient.

The system was implemented on Xilinx Spartan ®-3A FPGA Starter. As the FPGA (Field-programmable gate array) set to the MicroBlaze Soft-Core ® Processor - Processor Embeded XILINX. During tests of the real system using a public database, and the results were compared with other works using other techniques for the diagnosis of a patient. Other factors were also seen, the implementation on dedicated hardware, which make this system more efficient by using the techniques employed.

Sumário

Capítulo 1 1

1.1. Considerações Gerais 1
1.2. Apreciações Sobre o Tema e a Justificativa do Trabalho 3
1.3. Problema do Trabalho 5
1.4. Objetivos 6
1.5. Metodologia 6
1.6. O Eletrocardiograma 7
1.7. Estrutura do Trabalho 10

Capítulo 2 12

2.1. Sistemas Embarcados 12
2.2. FPGAs 13
2.3. *SOFT CPU CORE* 14
2.4. Xilinx Microblaze 15

Capítulo 3 18

3.1. Considerações Iniciais 18
3.2. Sistema de Pré-Processamento do Sinal 18
3.3. Filtro 19

Capítulo 4 24

4.1. Considerações Gerais 24

4.2. Função *Membership* 24
4.3. Criação de Regras 40
4.4. O Processo *Fuzzy Clustering* 43
4.5. Aplicações utilizando o Método *Fuzzy Clustering* 48

Capítulo 5 50

5.1. Considerações iniciais 50
5.2. Resultado das Correlações 51

Capítulo 6 54

6.1. Considerações iniciais 54
6.2. Desenvolvimento do *Software* 54
6.3. Desenvolvimento do *Hardware* 56
6.3.1 Escolhendo as Derivações a Serem Recebidas 57
6.3.2 Recebendo o Sinal Amostrado 58
6.3.3 Saídas do Sistema 60
6.3.4 Hardwares para testes e verificações 63

Capítulo 7 65

7.1. Banco de Dados 65
7.2. Testes Realizados 65
7.3. Resultados Obtidos 66

Capítulo 8 69

Referências Bibliográficas 71

Lista de Figuras

Figura 1.1 – Blocos de aquisição e filtragem. 2
Figura 1.2 – Blocos das técnicas de *Fuzzy Clustering* e correlação. 3
Figura 1.3 – Sistema de condução do coração [20]. 8
Figura 1.4 – Derivações bipolares I, II, III [20]. 8
Figura 1.5 – Derivações unipolares aVF, aVL e aVR [20]. 9
Figura 1.6 – Derivações precordiais V1, V2, V3, V4, V5 e V6 [20]. 9
Figura 1.7 – Sinal característico de um eletrocardiograma. 10
Figura 2.1 – FPGA Utilizada para o desenvolvimento do sistema. 14
Figura 2.2 – Características do núcleo do Microblaze [23]. 15
Figura 2.3 – Fluxo de projeto EDK da Xilinx. 16
Figura 2.4 – Placa XILINX SPARTAN 3A STARTER KIT. 17
Figura 3.1 – Sistema de Pré-processamento. 19
Figura 3.2 – Função de transferência de um filtro Butterworth [24]. 20
Figura 3.3 – Inclinação de atenuação do filtro Butterworth em função da variação de sua ordem [24]. 21
Figura 3.4 – Gráfico da resposta em frequência do filtro utilizado [24]. 21
Figura 3.5 – Exemplo de um sinal de eletrocardiograma **antes** de passar pelo bloco de filtragem e eliminação do nível DC 22
Figura 4.1 – Exemplos para comparação de velocidades. 25
Figura 4.2 – Exemplo de função *membership* brusca. 26
Figura 4.3 – Função *membership* suave. 26
Figura 4.4 – Função *membership* triangular. 27
Figura 4.5 – Função *membership* sigma. 28
Figura 4.6 – Função *membership* Gaussiana. 29
Figura 4.7 – Função *membership* formada pela diferença de duas funções sigma. 30
Figura 4.8 – Função *membership* sino generalizada 31
Figura 4.9 – Função sen($x_1.x_2$). 32
Figura 4.10 – Gráfico *membership* sino. 33
Figura 4.11 – Gráfico gerado pelo programa para *membership* sino. 34

Figura 4.12 – Gráfico do erro gerado entre o sinal esperado para o sistema e o sinal gerado do sistema. 34
Figura 4.13 – Gráfico da função sen($x_1.x_2$). 35
Figura 4.14 – Gráfico *membership* Gauss. 36
Figura 4.15 – Gráfico de saída do programa para *membership* Gauss. 36
Figura 4.16 – Gráfico do erro gerado entre a função sen($x_1.x_2$) e a função gerada pelo programa, utilizando *membership* Gauss. 37
Figura 4.17 – Gráfico da função sen($x_1.x_2$) a ser gerada pelo programa. 38
Figura 4.18 – Gráfico *membership* triangular. 38
Figura 4.19 – Gráfico de saída para *membership* triangular. 39
Figura 4.20 – Gráfico do erro gerado pelo programa entre a função desejada e a função a ser gerada. 40
Figura 4.21 – Função que deve ser gerada pelo programa. 44
Figura 4.22 – Funções de *memberships* utilizadas pelo programa. 45
Figura 4.23 – Função gerada na saída do programa. 45
Figura 4.24 – Gráfico do erro gerado pelo programa entre a função esperada e a função gerada. 46
Figura 4.25 – Gráfico que mostra a posição de cada cluster criado pelo programa. 47
Figura 4.26 – Gráfico que mostra a posição de cada cluster e a posição de cada ponto de saída do programa. 47
Figura 4.27 – Sinal de eletrocardiograma após pré-processamento. 48
Figura 4.28 – Clusters gerados pelo processo de *Fuzzy Clustering*. 49
Figura 5.1 – Sinais em análise. 52
Figura 5.2 – Sinais com forte correlação direta. 53
Figura 5.3 – Sinais com forte correlação inversa. 53
Figura 6.1 – Diagrama de blocos do programa. 55
Figura 6.2 – Configuração da entrada das derivações. 57
Figura 6.3 – Exemplo das chaves selecionando a derivação V4. 58
Figura 6.4 – Parâmetros necessários para a aquisição das amostras. 59
Figura 6.5 – Enviando o sinal para o sistema. 60
Figura 6.6 – Identificação dos *LEDs*. 60
Figura 6.7 – Derivações exibidas nos LED*s* (1º momento). 62
Figura 6.8 – Diagnóstico exibido pelos *LEDs* (2º momento), neste caso angina. 62
Figura 6.9 – Indicação das cardiopatias pelos *LEDs*. 63
Figura 6.10 – Saída da porta RS-232. 64
Figura 7.1 – Gráfico da quantidade de *clocks* por amostra. 68

Lista de Tabelas

Tabela 6.1 – Hardware Utilizado no Sistema. 56
Tabela 6.2 – Classificação das Derivações. 57
Tabela 6.3 – LEDs Ativos pela derivação escolhida. 61
Tabela 6.4 – Diagnóstico final do sistema. 61
Tabela 7.1 – Comparação de resultados 67

Lista de Símbolos

Hz	Hertz
N	Ordem do Filtro Butterworth
w	Frequência do filtro Butterworth
w_p	Frequência de corte do filtro Butterworth

Capítulo 1

Introdução

1.1. Considerações Gerais

De acordo com o governo norte-americano, através do *National Center for Health Statistics* [1] a principal causa de morte em humanos se deve a doenças do coração, superando até mesmo as mortes ocasionadas devido ao câncer. Mais de 910 mil americanos morrem anualmente em consequência de doenças do coração e, mais de 70 milhões vivem com alguma forma de doença do coração como pressão alta, derrame e angina [2]. Visando a redução desses números, várias pesquisas são desenvolvidas buscando formas de tornar o diagnóstico mais rápido, preciso e com uma antecedência que permita, de alguma forma, elevar as chances de sobrevivência dos pacientes, através de tratamento específico para uma determinada cardiopatia.

Os sinais cardíacos dos pacientes são obtidos através de um equipamento denominado eletrocardiógrafo (galvanômetro). Este por sua vez apresenta esses sinais na forma do eletrocardiograma (ECG). Denomina-se eletrocardiograma de superfície os sinais elétricos de origem cardíaca captados em pontos particulares do corpo humano. Ele é um registro das variações dos potencias elétricos gerados pela atividade elétrica do coração. O diagnóstico é baseado na extração de informações dos picos de ondas e intervalos de tempo do sinal do ECG. O ECG é um método não invasivo (externo ao corpo humano) seguro, reproduzível, de fácil obtenção, baixo custo e que fornece importantes apontamentos para

análises e diagnósticos de anomalias cardíacas, na opinião da equipe de especialistas da Sociedade Brasileira de Cardiologia [3].

O ECG é a representação de um sinal analógico tendo como grandeza no eixo das abscissas o tempo, normalmente, em segundos, e no eixo das ordenadas, volts (mV). Desta forma antes de sua análise é necessário um pré-processamento, utilizando filtros. Este pré-processamento é necessário devido à presença de ruídos indesejados misturados com os sinais cardíacos. Estes ruídos são oriundos, principalmente, devido à presença de outros equipamentos próximos ao eletrocardiógrafo e ao movimento dos músculos do paciente. Na Figura 1.1 são mostrados todos os blocos utilizados que representam esta etapa.

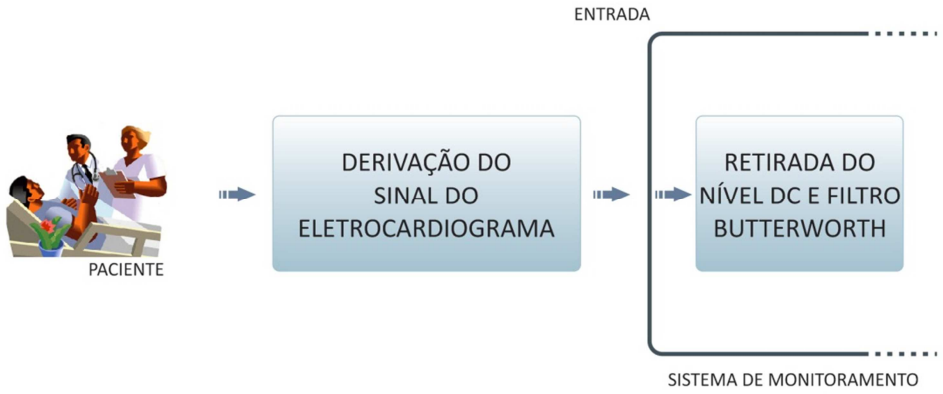

Figura 1.1 – Blocos de aquisição e filtragem.

O primeiro bloco representa as derivações. Estas derivações são sinais obtidos pelos sensores, posicionados em pontos estratégicos do corpo humano para captar os sinais elétricos gerados pelo coração. Normalmente, para uma maior certeza no diagnóstico, são analisadas 12 derivações do eletrocardiograma, detalhadas no item 1.6 deste capítulo, porém, sabe-se que com apenas duas derivações é possível gerar diagnósticos [4]. No segundo bloco está o filtro que irá eliminar os ruídos do sinal analisado. Neste bloco também é feita a retirada do nível DC presente no sinal.

Na Figura 1.2 são apresentados os blocos, que representam a segunda etapa, objeto deste trabalho. No primeiro bloco é feito o processo *Fuzzy Clustering*, onde são obtidos os pontos que descrevem as principais características do sinal do eletrocardiograma e no segundo bloco, com a ajuda de um banco de dados de derivações de sinais de eletrocardiograma conhecidos, é feita a correlação do sinal em análise com os sinais do banco de dados, obtendo desta forma o diagnóstico do paciente.

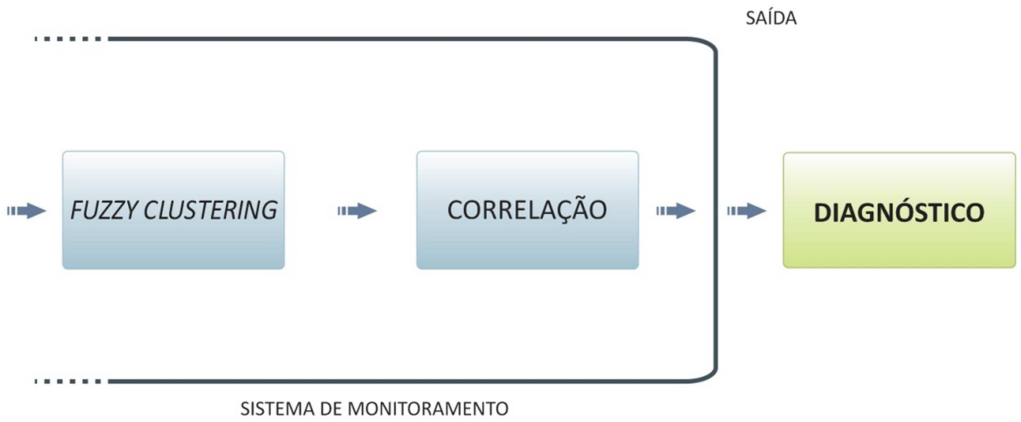

Figura 1.2 – Blocos das técnicas de *Fuzzy Clustering* e correlação.

Tendo em vista a importância do desenvolvimento de tecnologias na área cardíaca descritos anteriormente, este trabalho apresenta um sistema embarcado que utiliza uma técnica de *Fuzzy Clustering* para obter características de um sinal de eletrocardiograma, mostrando sua aplicação, resultados de comparações com outras técnicas de processamento de sinais já estudadas, avaliações de desempenho e custo em hardware dedicado.

1.2. Apreciações Sobre o Tema e a Justificativa do Trabalho

As primeiras tentativas de automatizar a interpretação de ECG por computador (separação automática de ECGs normais de ECGs anormais feito por um programa de computador concebido para reconhecer e medir os componentes de onda do sinal de ECG digitalizados) foram feitas por Pipberg no final da década de 1950 [5-7]. Os primeiros programas disponíveis comercialmente foram introduzidos no início de 1970 [5] [7]. Sistemas de interpretação automática do ECG têm se tornado mais sofisticados, com um menor custo e um número crescente de programas e técnicas disponíveis. O aumento contínuo no uso de sistemas computacionais para interpretação do ECG leva a necessidade de avaliar parâmetros como o seu desempenho, precisão nos diagnósticos gerados e a compreensão de suas vantagens e desvantagens.

As pesquisas sobre o uso de sistemas computacionais que utilizam técnicas diversas para a geração de diagnósticos cardíacos abrangem várias áreas do conhecimento. Como exemplo, pode-se citar o processamento de sinais, e a aplicação de filtros e algoritmos de inteligência artificial (IA) para identificação e classificação destes sinais. A maioria das pesquisas nessa área tem como objetivo fornecer meios automáticos de diagnósticos, ou a

melhor preparação das etapas necessárias para este fim, como forma de suporte a médicos e outros profissionais da saúde.

No estudo de processamento de sinais, um dos tópicos a serem abordados são os filtros. Nianqiang, Yongbing e Guoyi [8] citam a importância da aplicação de filtros digitais em sinais de ECG visando a eliminação dos ruídos e reforçando a necessidade deste tipo de processamento ser feito em tempo real e em alta velocidade.

A aplicação de algoritmos de IA em problemas cardíacos é bastante utilizada. Por exemplo, Mitsukura [9] cita como uma das principais motivações e necessidades da pesquisa de um sistema médico de diagnósticos, a alta mortalidade causada por doenças do coração. Pode-se citar duas etapas neste processo: a detecção e a classificação dos sinais ECG. Yan [10] apresenta um algoritmo matemático para identificação do complexo QRS, detalhado ainda neste capítulo, com uma alta precisão. Esta identificação é importante pois é a base para a classificação de doenças cardíacas.

Estando o sinal ECG devidamente filtrado e identificado faz-se a classificação que poderá demonstrar, ou não, o reconhecimento de alguma cardiopatia. Jewajinda e Chongstitvatana [11] demonstram a classificação do sinal do ECG para monitoramento de pacientes por um longo prazo.

Para obter o provável diagnóstico do sinal do ECG é utilizada a técnica da correlação. Neste caso é feita uma comparação do ECG em análise com ECGs da base de dados contendo as cardiopatias já definidas por uma equipe médica. Para obter o grau de eficiência dos algoritmos demonstrados neste trabalho são feitas comparações com outros trabalhos que possuem os mesmos parâmetros estatísticos de acertos ou erros nos diagnósticos, destacando-se Andreão [12], Jager [13], Maglaveras [14], Taddei [15] e Vila [16]. Estes apresentam sistemas computacionais que utilizam técnicas de processamento de sinais para interpretação de características de um eletrocardiograma, o que permite obter diagnósticos prévios de alguma cardiopatia em um paciente. Com estas técnicas torna-se possível o diagnóstico automático do paciente, permitindo, por exemplo, um monitoramento remoto, onde o acionamento de um alarme, em tempo real, avisaria um paciente ou até mesmo uma equipe médica, que pode tomar providências com antecedência evitando que o problema só seja tratado em um estágio mais grave, aumentando as chances de sobrevivência do mesmo.

Depois da validação dos algoritmos no ambiente de software, utilizando simuladores, como por exemplo, o Matlab® no caso do *Fuzzy Clustering* algorithm, o objetivo de vários trabalhos é o desenvolvimento de um sistema embarcado, em FPGA que

faça diagnósticos ou parte do processo para sua obtenção. Armato [17] exemplifica a extração dos complexos QRS, classificação em tempo real entre normal ou patológica e a integração em FPGA. Portanto devido a redução de custos, observados nos sistemas embarcados nos últimos anos, uma tendência natural de validação e aplicação é em sistemas que contenham FPGAs devido a facilidade e maior velocidade de implementação.

1.3. Problema do Trabalho

O problema a ser investigado é a possibilidade de aplicação e implantação de um sistema real, ou seja, um equipamento, que utiliza uma técnica de processamento de sinais para a obtenção, com maior exatidão e simplicidade, das características mais relevantes de um sinal de eletrocardiograma. Com objetivo de execução em um sistema embarcado de baixa frequência, para os padrões atuais (50MHz), consequentemente um hardware mais simples e barato para obtenção de diagnósticos.

Para a verificação da eficácia do trabalho proposto, foi adquirido um banco de dados de sinais de eletrocardiogramas reais, onde cada sinal apresenta as principais características de um eletrocardiograma com uma determinada cardiopatia. O banco de dados utilizado foi obtido através do PhysioNet (www.physionet.org). PhysioNet é um projeto cooperativo iniciado pelo *Boston's Beth Israel Deaconess Medical Center/Harvard Medical School, Boston University, Mc Gill University e Massachustts Institute of Technology (MIT)*, que consiste em um serviço público para pesquisa de sinais fisiológicos complexos, financiado pelo *National Center for Research Resources* e pelo *National Institutes of Health*. O PhysioNet oferece livre acesso via Internet a diversas bases de dados de sinais fisiológicos e softwares relacionados à leitura das mesmas [18].

Devido à presença de ruídos e de componentes DC, foi utilizado um sistema de pré-processamento que é composto de um filtro Butterworth de terceira ordem, implementado como séries numéricas, ou seja, em algoritmos computacionais, e um processamento para eliminar o nível de sinal DC presente junto com os sinais em análise, permitindo assim maior precisão nos resultados obtidos.

1.4. Objetivos

O objetivo geral deste trabalho é apresentar um sistema embarcado, implantado em hardware dedicado, que tem como tarefa receber um sinal de eletrocardiograma e fazer todo o processamento até gerar um, possível, diagnóstico. A implantação do sistema pode ser dividida em aquisição dos sinais, tratamento dos sinais, filtragem, *Fuzzy Clustering* e correlação, visando sempre um sistema de diagnósticos rápido, com precisão nos resultados, que tenha um baixo custo e algoritmos portáveis. Os objetivos específicos, que derivam do objetivo geral, são:

- Dominar e validar todos os algoritmos em hardware;
- Implementar um sistema de aquisição e preparação dos sinais;
- Dominar o processo de *Fuzzy Clustering*;
- Adaptar a técnica de *Fuzzy Clustering* para a aplicação desejada;
- Demonstrar as vantagens na utilização do processo de *Fuzzy Clustering* em hardware, em relação à diminuição no número de sinais amostrados;
- Realizar testes, utilizando a correlação, com sinais de eletrocardiogramas reais para a comprovação da eficácia do sistema utilizado.

1.5. Metodologia

O primeiro passo foi a realização de uma pesquisa bibliográfica, com o objetivo de ampliar conhecimentos para verificação do “Estado da Arte” da pesquisa sobre eletrocardiogramas, técnicas de processamento de sinais e sistemas embarcados.

Na segunda etapa foi planejado o algoritmo para aquisição e pré-processamento de dados pelo hardware.

A terceira etapa da pesquisa consistiu no desenvolvimento de algoritmos mais eficientes, para implantação em hardware, utilizando o processo de *Fuzzy Clustering*.

Na fase seguinte, foram feitos testes com sinais de eletrocardiograma reais para comprovação da técnica utilizada e validação do hardware.

1.6. O Eletrocardiograma

O eletrocardiograma é o registro de todas as atividades elétricas originadas pelo coração. As informações obtidas através do eletrocardiograma são de grande importância para a determinação de algum distúrbio que possa estar ocorrendo no funcionamento ou na estrutura do coração [19].

O eletrocardiograma é obtido, através de um aparelho chamado de galvanômetro, que mede a corrente entre dois eletrodos colocados em pontos estratégicos do corpo humano [19].

No corpo humano, a célula muscular em repouso, polarizada, é rica em potássio. O meio externo a esta célula é rico em sódio. Desta maneira, a célula em repouso polarizada é mais negativa em relação ao meio externo que é mais positivo em relação à célula [19].

Quando a célula é ativada ocorre a despolarização celular onde o interior da célula passa a ficar mais positivo em relação ao meio externo. Após a repolarização, a célula volta às suas condições normais iniciais [19].

O sistema de condução do coração nada mais é do que um processo de despolarização e repolarização celular. Este sistema é formado pelo nódulo sinusal (NS), feixes internodais da junção atrioventricular (NAV), feixe de His, ramos do feixe e pela rede de Purkinje como indicado na Figura 1.3 [19].

O nodo sinusal se encontra na parede posterior do átrio direito e comanda o ritmo cardíaco. Ele é nutrido pela artéria do nódulo sinusal. A despolarização iniciada no nódulo sinusal se propaga pelo sistema de condução sinoatrial ativando o endocárdio e o epicárdio dos átrios [19].

Os feixes internodais ligam o nódulo sinusal ao nódulo átrio ventricular. O feixe internodal anterior fornece um ramo que chega até o átrio esquerdo [19].

O nódulo átrioventricular recebe fibras do miocárdio atrial e de suas margens anterior e inferior emerge o feixe de His [19].

No feixe de His origina o ramo direito e o ramo esquerdo. Estes continuam por uma rede subendocárdica de fibras de Purkinje [19].

O ventrículo esquerdo eletricamente é mais potente que o ventrículo direito, logo a ativação caminhará para esquerda e para trás [19].

A repolarização atrial não é registrada no eletrocardiograma porque ela ocorre ao mesmo tempo que a despolarização ventricular, que é um processo elétrico mais potente [19].

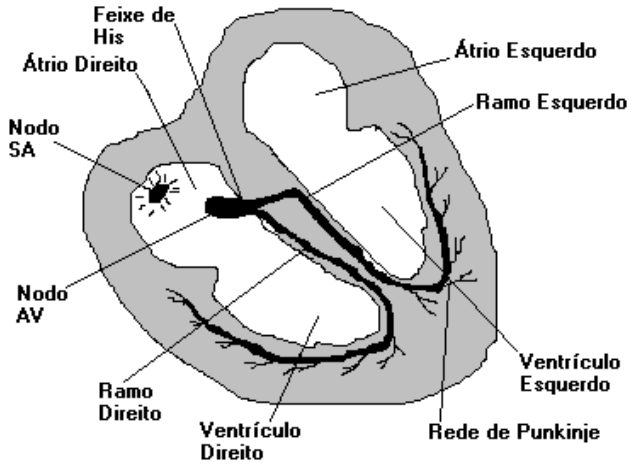

Figura 1.3 – Sistema de condução do coração [20].

Devido a estes fenômenos elétricos gerados pelo coração, é possível medir diferenças de potencial na superfície do corpo humano. Dependendo dos locais no corpo humano onde os eletrodos do aparelho de medida são colocados, são caracterizadas as derivações correspondentes de cada sinal [20]. As derivações mais utilizadas são:

- Derivações bipolares, que são derivações obtidas pela utilização de dois eletrodos posicionados no corpo humano, como indicado na Figura 1.4 [20].

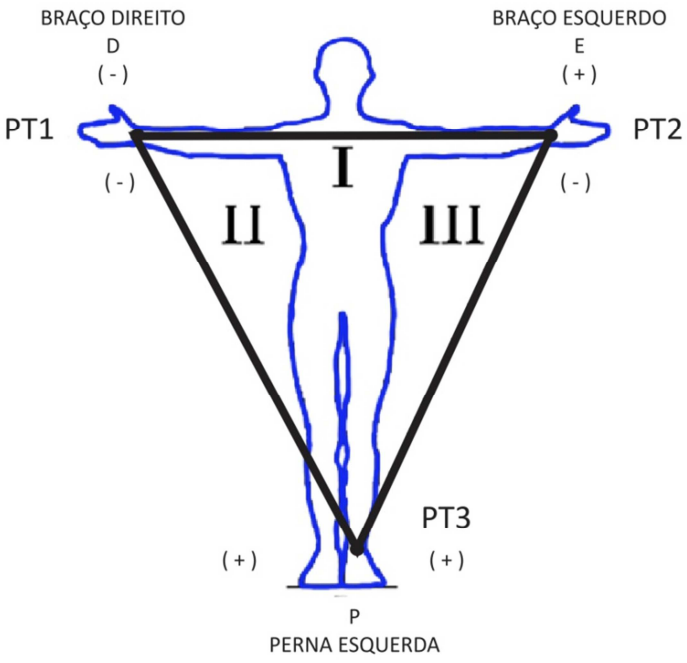

Figura 1.4 – Derivações bipolares I, II, III [20].

- Derivações unipolares, onde um eletrodo é colocado em um dos pontos citados na derivação bipolar e o outro eletrodo é conectado em um terminal central, como mostrado na Figura 1.5 [20].

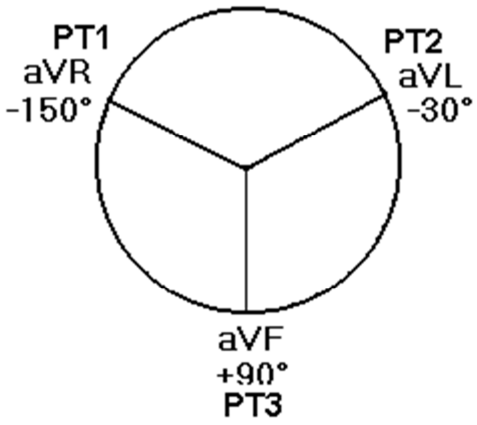

Figura 1.5 – Derivações unipolares aVF, aVL e aVR [20].

- Derivações precordiais, estas derivações são obtidas através da análise que utiliza uma visão de um plano chamado de plano horizontal, que seria uma vista de cima do corpo humano. A Figura 1.6 mostra a posição destas derivações [20].

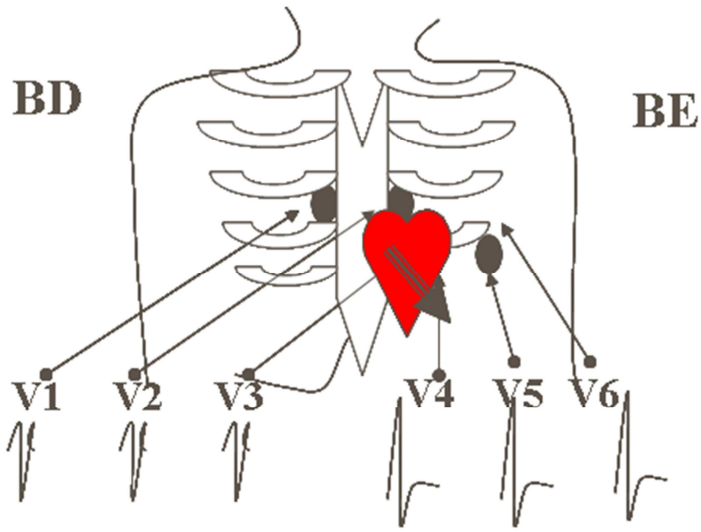

Figura 1.6 – Derivações precordiais V1, V2, V3, V4, V5 e V6 [20].

De acordo com estas derivações é possível se obter sinais que podem indicar alguma anomalia no funcionamento do coração. Quanto maior o número de derivações utilizadas mais preciso pode ser o diagnóstico.

Na Figura 1.7 é apresentado um sinal característico de um eletrocardiograma com seus principais trechos. Cada trecho representa uma etapa do processo de despolarização e repolarização celular.

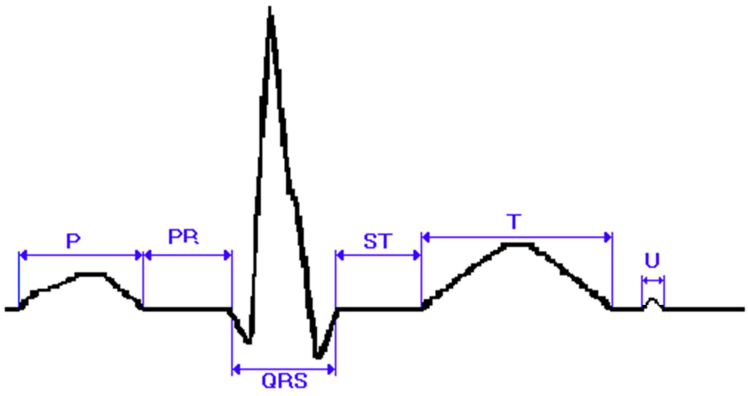

Figura 1.7 – Sinal característico de um eletrocardiograma.

O trecho P representa o início da despolarização no átrio direito e depois no átrio esquerdo. O segmento PR descreve o momento em que o impulso elétrico caminha pelo feixe de His e pelos ramos direito e esquerdo. A despolarização dos ventrículos é representada pelo segmento QRS. O segmento ST representa a fase inicial da repolarização dos ventrículos e a onda T descreve a repolarização final dos ventrículos. Após a onda T têm-se a onda U. Esta onda é considerada normal quando positiva e quando negativa indica uma anomalia. Esta onda não é constante [19].

1.7. Estrutura do Trabalho

O Capítulo 2 discorre sobre *Soft CPU Core*, e FPGAs, apresenta o sistema Microblaze® Soft-Core Processor - XILINX Embeded Processor, e especifica sua tecnologia e funcionalidades utilizadas neste trabalho.

O Capítulo 3 apresenta o sistema de aquisição e pré-processamento bem como uma breve explanação sobre o filtro utilizado neste sistema e o processo de eliminação do nível DC.

O Capítulo 4 aborda os conceitos que envolvem o processo de *Fuzzy Clustering* e mostra alguns exemplos e aplicações, utilizando este processo.

O Capítulo 5 será apresentado o conceito sobre correlação e exemplos sobre a sua utilização.

O Capítulo 6 demonstrará algumas características e o funcionamento do sistema no hardware e software desenvolvidos.

No Capítulo 7 observam-se os testes realizados, os resultados obtidos e comparações com outras técnicas utilizadas em outros trabalhos.

As conclusões e considerações finais encontram-se dispostas no Capítulo 8.

Capítulo 2

O *Hardware* Utilizado e Definições Necessárias

2.1. Sistemas Embarcados

Conforme apresentado no Capítulo 1, um dos objetivos é a implantação de técnicas de processamento de sinais, no caso a *Fuzzy Clustering* e a correlação, utilizando um sistema embarcado.

Sistemas embarcados são sistemas responsáveis por um conjunto restrito de funções especificas e correlacionadas. Eles são formados basicamente pelos mesmos componentes de uma arquitetura de computador: microprocessador, memória principal, interfaces e outros, dependendo do sistema. [21]. Existem algumas limitações de hardware, em comparação com uma arquitetura de computador, como por exemplo, frequência de processamento, memória, dentre outros, que, em geral, são bem menores que em sistemas computacionais maiores. Por outro lado as vantagens, nos sistemas embarcados em relação aos computadores, são na maioria dos casos: custo, tamanho, peso, consumo e especificidade. Um sistema embarcado pode consistir de um ou vários circuitos integrados [21].

Em aplicações onde um circuito integrado personalizado é essencial, existe a opção de usar circuitos integrados programáveis, chamados de FPGAs (Field-Programmable Gate Arrays) ou de LCAs (Logic-Cell Arrays).

2.2. FPGAs

As FPGAS são circuitos integrados (CIs) com características de serem programáveis. A Tecnologia foi criada pela XILINX Inc., e teve o seu lançamento no ano de 1985. Esses circuitos são compostos por um enorme número de chaves programáveis, que podem ser configurados para simular o comportamento de qualquer outro circuito digital. Para programar esses circuitos podem ser utilizadas linguagens de descrição de hardware – HDL (Hardware Description Language). As HDLs mais usuais atualmente são VHDL, Verilog e SystemVerilog.

A grande vantagem da FPGA em relação aos CIs comuns é a capacidade de possuir um desempenho de hardware com a flexibilidade de um software [22]. Entretanto as FPGAs são naturalmente mais caras que CIs convencionais, porém são uma boa opção em situações onde são necessários apenas alguns circuitos exclusivos.

Existem diversas empresas fabricantes de FPGAs, destacando-se XILINX e ALTERA. O sistema foi implementado em uma FPGA XILINX da família SPARTAN, conforme ilustra a figura 2.1.

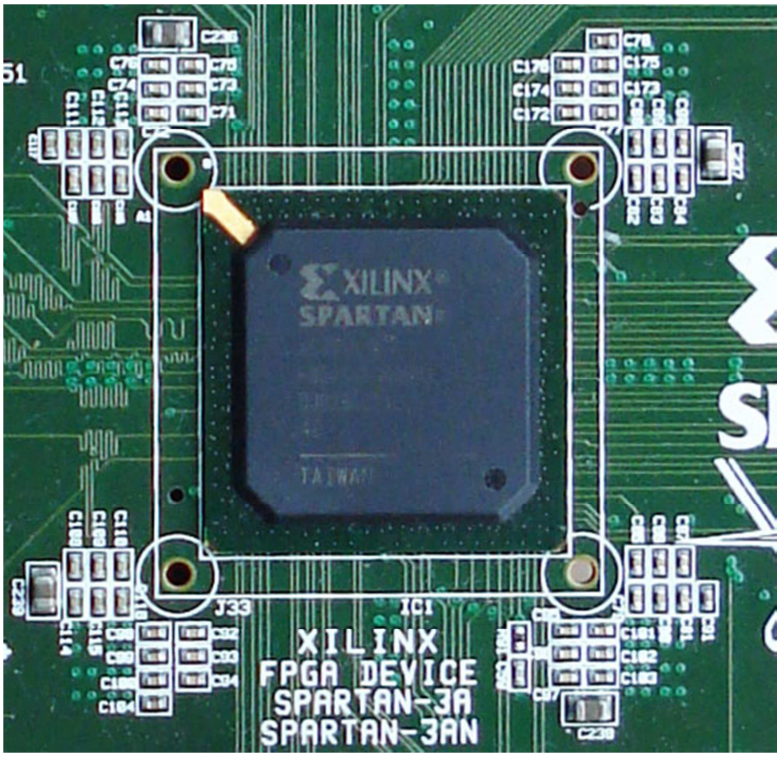

Figura 2.1 – FPGA Utilizada para o desenvolvimento do sistema.

2.3. *SOFT CPU CORE*

Em muitas aplicações é conveniente possuir um microprocessador e uma FPGA, que podem estar separados. Mas eles também podem estar combinados em um único CI, proporcionando menos consumo de energia, arquitetura de placas mais simples, e menos problemas com a integridade do sinal e interferência eletromagnética.

Um *Soft CPU Core*, processador "leve", é uma Propriedade Intelectual - IP (Intellectual Property) da Xilinx, que é implementado usando as lógicas da FPGA. Dentre os principais benefícios do uso de um processador *soft core* destacam-se a relação custo por benefício entre preço e desempenho, além de simplicidade de projeto, menor tempo de produção, fácil integração com a FPGA e outras tecnologias, e devido às atualizações no IP, vida útil longa e alta reusabilidade [23]. Normalmente os "*Soft CPU*" são escritos em linguagens de descrição de hardware – HDL, e não utilizam partes específicas da FPGA. Assim não dependem da tecnologia e são flexíveis, apesar de serem mais lentos que os "*Hard CPU Core*".

Os *Hard CPU Cores*, processador "pesado", são desenvolvidos em baixo nível (nível de silício). Estes são especializados e muito pouco flexíveis, contudo são bem mais rápidos que os *Soft CPU Cores*.

2.4. Xilinx Microblaze

O Microblaze é um processador 32-bit RISC Soft Processor Core. Processadores RISC (*Reduced Instruction Set Computer*), ou computador com um conjunto reduzido de instruções, são capazes de executar apenas algumas poucas instruções simples. Justamente por isso, os chips baseados nesta arquitetura são mais simples e muito mais baratos. Pela sua forma de implementação, em FPGA, o Microblaze é tratado em algumas bibliografias como um processador virtual. Ele possui instruções otimizadas para aplicações embarcadas [23] e trabalha, por padrão, com vários tipos de memórias SDR, DDR, DDR2, SRAM e Flash. Possui também uma unidade de ponto flutuante, FPU, compatível com IEEE-754 de precisão simples [23]. A FPU é de extrema importância para este trabalho. A Figura 2.2 de [23] ilustra as características do núcleo do processador.

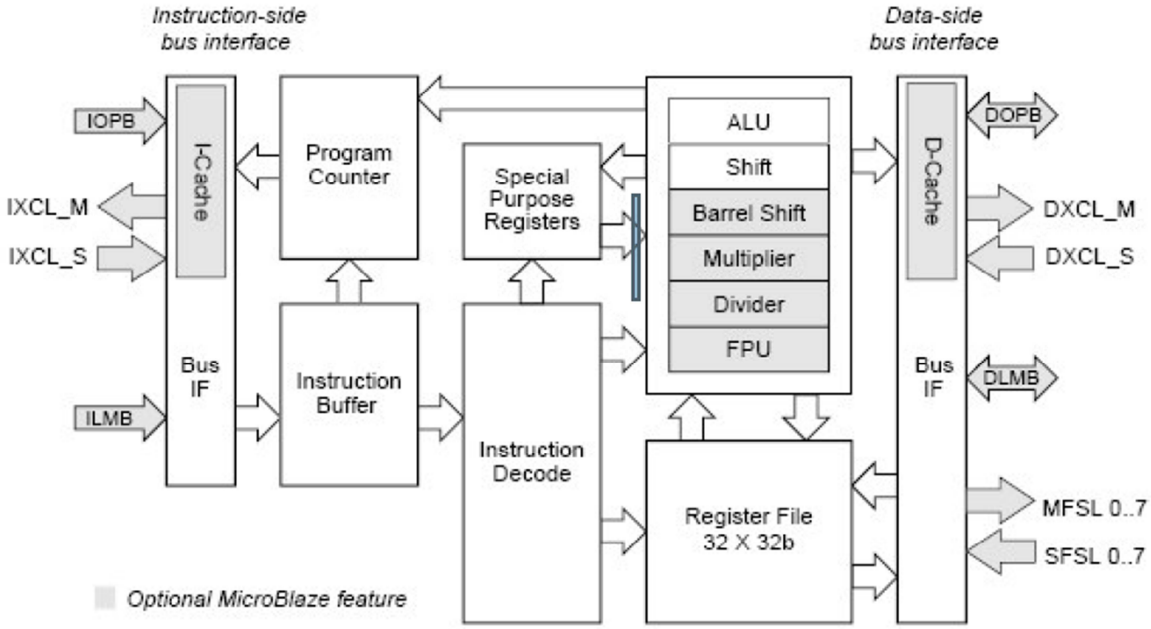

Figura 2.2 – Características do núcleo do Microblaze [23].

A arquitetura básica consiste em trinta e dois registradores de propósito-geral, uma Unidade Lógica Aritmética (ULA), uma unidade de deslocamento, e dois níveis de interrupção. Para a placa de desenvolvimento XILINX SPARTAN 3A Starter kit, usada no trabalho, a frequência de *clock* interna máxima do Microblaze é de 50 MHz.

A suíte de desenvolvimento deste sistema é a XILINX *Software Development Kit* - SDK, que divide o projeto em duas partes: projeto de hardware e projeto de software.

O desenvolvimento de programas, ou projeto de software, para este processador deve ser feito em linguagem C ou C++, pois estas foram escolhidas pela XILINX. Portanto a suíte de desenvolvimento para este processador, XILINX Software Development Kit - SDK, possui compiladores que geram o código de máquina, a partir de programas em C ou C++, necessário para a execução no Microblaze. Neste trabalho a linguagem de programação utilizada é C.

O XILINX Plataform Studio - XPS, é a suíte responsável pela geração de todos os IP cores do MicroBlaze e seus respectivos periféricos, ou seja, o projeto de hardware. O XPS possui uma biblioteca com algumas arquiteturas de hardware escritas, comumente, em VHDL. Esta suíte faz a junção e compilação de todos os códigos VHDL dos componentes escolhidos e implanta-os na FPGA. O fluxo de projeto é ilustrado na Figura 2.3.

Figura 2.3 – Fluxo de projeto EDK da Xilinx.

Os componentes, de hardware, necessários para este trabalho são:

- Microblaze a 50 MHz de clock com FPU;
- 32 MB de memória ram DDR2;
- Interface serial (RS232);
- Chaves (Dip Switches);
- LEDs;
- Botões;
- Contador de ciclos de *clock* (IP Clock Counter).

A Figura 2.4 ilustra a placa XILINX Spartan 3A Starter Kit com os componentes necessários indicados, destacando-se:

1 – FPGA;

2 – Chaves;

3 – *Leds;*

4 – Botões;

5 – Porta RS232;

6 – Entrada USB;

7 – Alimentação da placa.

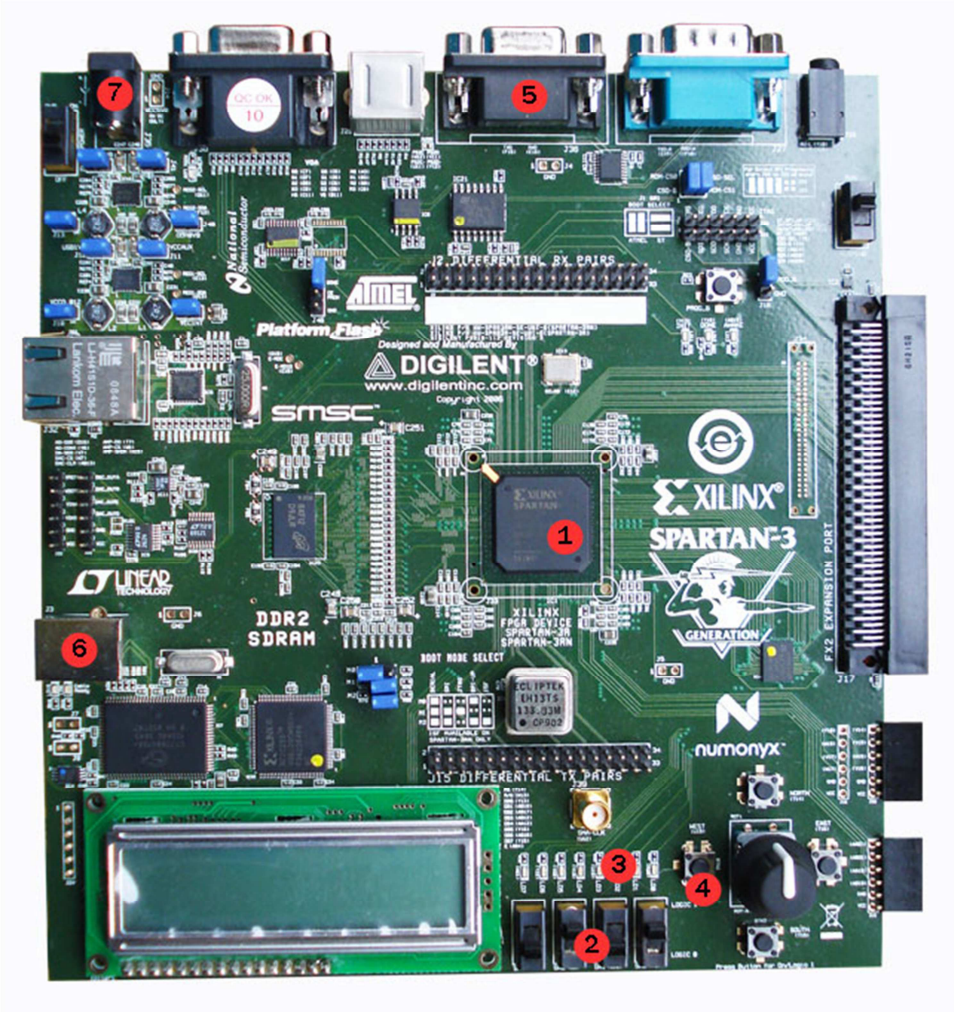

Figura 2.4 – Placa XILINX SPARTAN 3A STARTER KIT.

As formas de funcionamento do sistema de diagnóstico cardíaco, em integração com os periféricos e o microprocessador, serão detalhadas no Capítulo 6.

Capítulo 3

Fundamentos Teóricos

3.1. Considerações Iniciais

Conforme apresentado no Capítulo 1, o objetivo é aplicar uma ferramenta de processamentos de sinais, no caso o *Fuzzy Clustering* e a correlação, em um sistema embarcado para obter o provável diagnóstico de pacientes. Para se testar a eficácia do sistema, foi necessária a utilização de sinais de eletrocardiogramas reais. A forma específica da entrada desses dados no hardware é via porta serial RS-232, que será detalhada no Capítulo 6. Para as fundamentações teóricas seguintes consideram-se os dados do eletrocardiograma já inseridos no sistema. Devido à presença de ruídos, antes da análise do eletrocardiograma, foi utilizado um filtro e um processo de eliminação de qualquer nível DC presente neste sinal, ambos através de métodos numéricos com a utilização de algoritmos lógicos.

3.2. Sistema de Pré-Processamento do Sinal

Para a utilização de sinais reais de eletrocardiograma, foi necessário utilizar um sistema de pré-processamento para retirar o ruído de frequências mais altas do que as frequências de um eletrocardiograma.

O sistema de pré-processamento tem a função de preparar o sinal para que possa ser processado sem a interferência de sinais indesejados. A Figura 3.1 mostra o sistema de pré-processamento utilizado no sistema proposto.

Figura 3.1 – Sistema de Pré-processamento.

De acordo com a Figura 3.1, o primeiro bloco do sistema de pré-processamento representa a entrada do sistema. De acordo com o banco de dados utilizado para os testes e com os sinais reais de eletrocardiogramas, estes já foram fornecidos amostrados, ou seja, para este sistema o sinal de eletrocardiograma utilizado já estava amostrado com tempo de amostragem de 0,003s.

O segundo bloco representa a utilização de um filtro passa baixa, para a eliminação de ruídos de frequências mais altas. Neste projeto, após alguns testes no software MATLAB, foi utilizado um algoritmo que representa um filtro Butterworth, que será detalhado adiante.

O terceiro bloco representa a eliminação de nível DC presente nos sinais de eletrocardiograma. Foi aplicada esta técnica de processamento, também através de algoritmos, para a eliminação de nível DC.

3.3. Filtro

O filtro tem a finalidade de eliminar sinais de frequências indesejadas. No sistema proposto, verificou-se a necessidade da utilização de um filtro passa baixa para eliminar os ruídos que possuem frequência mais alta do que a frequência do eletrocardiograma. Estes ruídos presentes no eletrocardiograma causam erros devido a alterações nas características de variação do sinal em análise.

Utilizando a ferramenta MatLab e realizando alguns testes e ajustes verificou-se que, para esta aplicação, o filtro de Butterworth apresenta resultados satisfatórios e por isto foi o escolhido. Sabe-se que este filtro tem como principal característica uma resposta em frequência sem ondulações em sua banda passante e uma resposta nula fora da sua faixa de passagem. Outra característica deste tipo de filtro é que alterando sua ordem, somente a inclinação da atenuação é alterada [24]. A magnitude da função de ordem N com uma banda passante com frequência de corte Wp é dada por:

$$\left|T(jw)\right| = \frac{1}{\sqrt{1+e^2\left(\frac{w}{w_p}\right)^{2N}}} \tag{3.1}$$

Nota-se que para este filtro a máxima variação da banda passante ocorre na frequência de corte, ou seja, quando w = wp [24].

A Figura 3.2 mostra um exemplo de resposta em frequência de um filtro Butterworth.

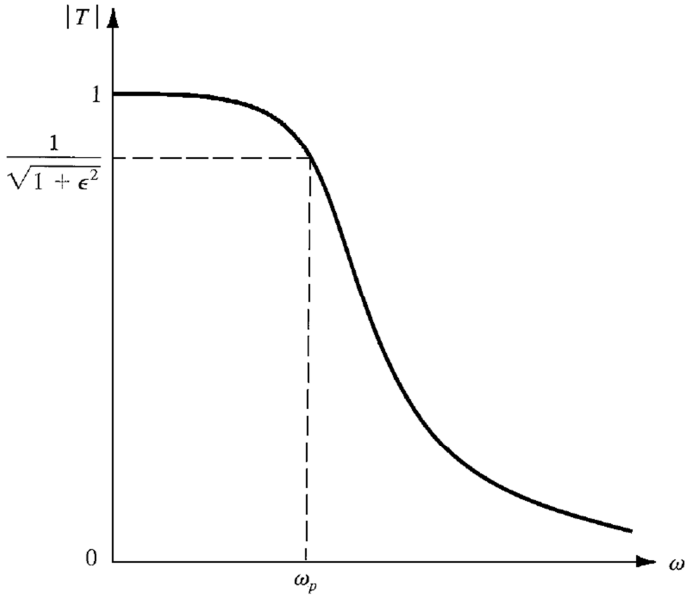

Figura 3.2 – Função de transferência de um filtro Butterworth [24].

A Figura 3.3 apresenta a atuação do filtro Butterworth de acordo com a variação da ordem do filtro.

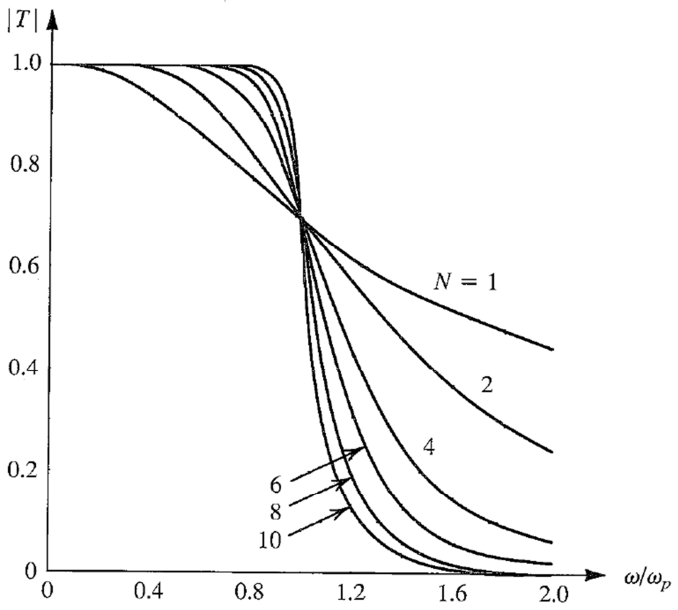

Figura 3.3 – Inclinação de atenuação do filtro Butterworth em função da variação de sua ordem [24].

Na Figura 3.4 é apresentado o gráfico de resposta em frequência do filtro de Butterworth utilizado neste sistema. Este filtro foi ajustado em um filtro de terceira ordem e com uma frequência de corte de 13,8 Hz.

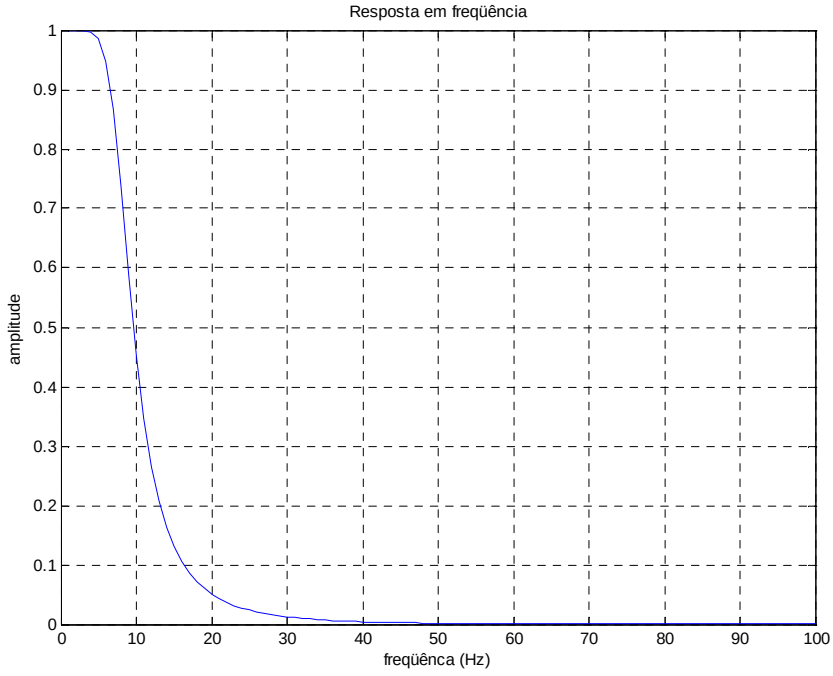

Figura 3.4 – Gráfico da resposta em frequência do filtro utilizado [24].

O sinal do eletrocardiograma, conforme o gráfico da Figura 3.5, é aplicado na entrada de um filtro Buttterworth. Este filtro elimina ruídos de frequências mais altas de tal

forma que em sua saída é obtido um sinal de eletrocardiograma apenas com as características de funcionamento do coração conforme ilustrado na Figura 3.6. Através de um processamento é feita a eliminação de nível DC presente no sinal, que para o método da correlação é fundamental visto que todos os sinais devem estar no mesmo nível. Após este processamento o sinal passa a ter as características mostradas no gráfico da Figura 3.6 e então está pronto para ser analisado.

Para a eliminação do nível DC, foi calculada a média aritmética das amostras do sinal em análise. O valor médio encontrado foi subtraído de cada amostra, e desta forma o sinal DC presente no eletrocardiograma foi eliminado, como observado na Figura 3.6.

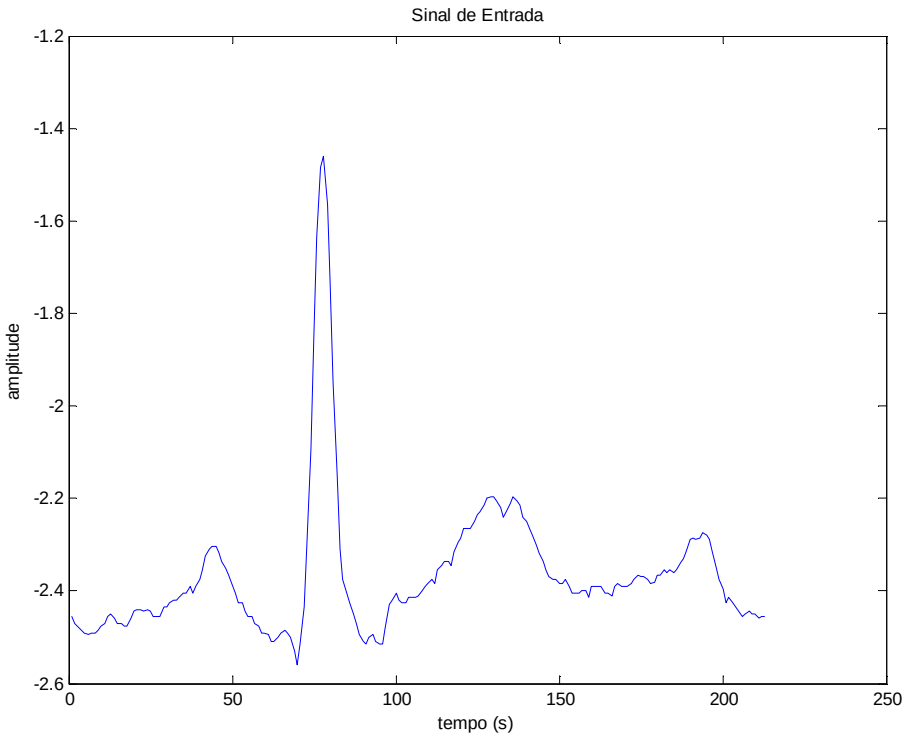

Figura 3.5 – Exemplo de um sinal de eletrocardiograma **antes** de passar pelo bloco de filtragem e eliminação do nível DC

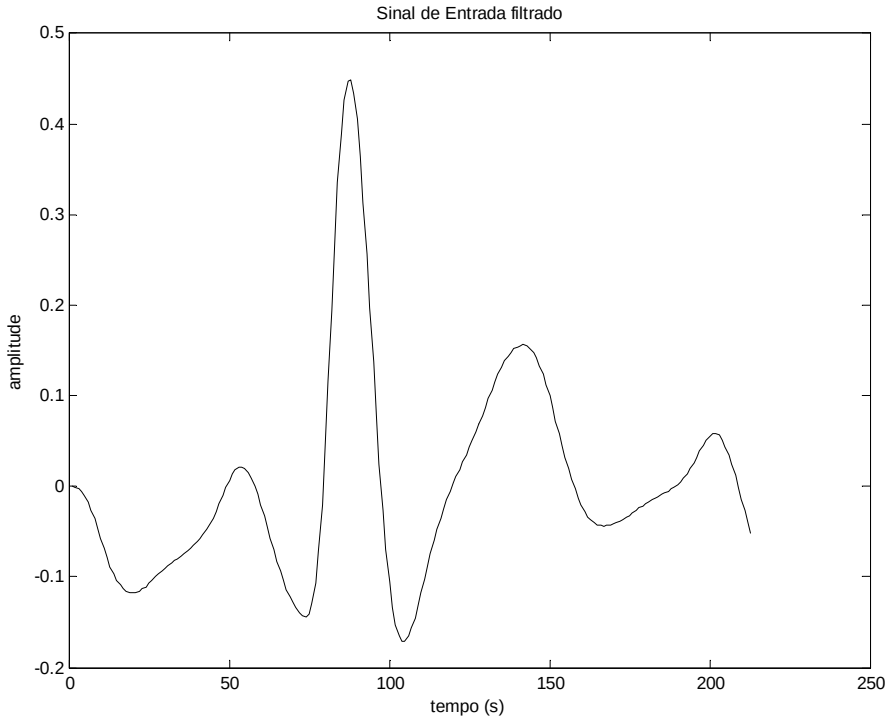

Figura 3.6 – Exemplo de um sinal de eletrocardiograma **após** passar pelo bloco de filtragem e eliminação do nível DC

Capítulo 4

Fuzzy Clustering

4.1. Considerações Gerais

Neste capítulo são apresentados todos os procedimentos realizados para a execução do processo de *Fuzzy Clustering*, que é a base para o funcionamento do sistema em hardware na qual se fundamenta a tese proposta.

O processo de *Fuzzy Clustering* pode ser dividido em algumas etapas:

- Primeiro deve-se escolher a função de membership a ser utilizada;

- Após a escolha da função de membership a ser utilizada, inicia-se a etapa da criação das regras. Cada regra descreve a tendência da saída do sistema para cada faixa de valores dos sinais de entrada. Estas regras descrevem as principais características de resposta do sistema a ser controlado.

4.2. Função *Membership*

Uma função de *membership* indica qual o grau de pertinência de um determinado elemento em relação a um evento. O valor do *membership* pode variar entre 0 e

1. Esta faixa de valores representa o grau de influência de um elemento em relação a um evento [25].

Para melhor compreender a definição desta função, imagine a necessidade de diferenciar um carro em alta velocidade e um carro com baixa velocidade. A Figura 4.1 ilustra cinco carros com suas respectivas velocidades:

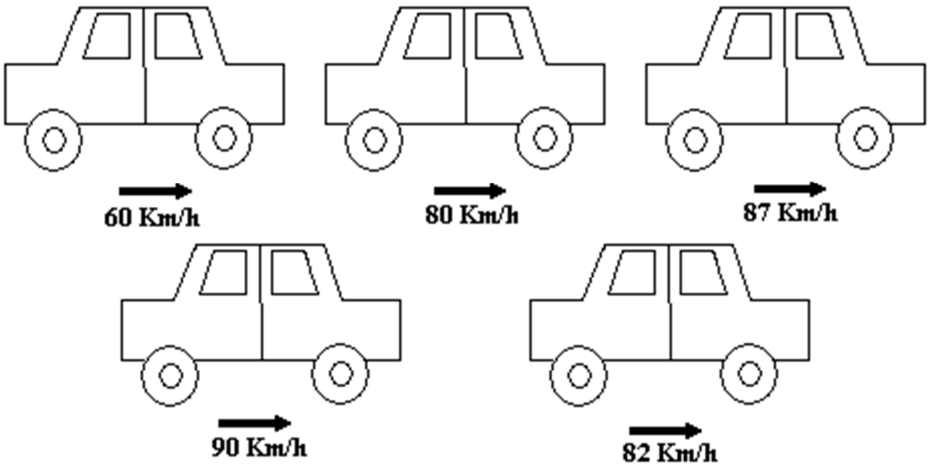

Figura 4.1 – Exemplos para comparação de velocidades.

Considerando-se que um carro a uma velocidade acima de 80 Km/h está acima da velocidade máxima permitida, os carros com a velocidade de 90 Km/h, 87 Km/h e 82 Km/h terão seus motoristas multados, pois estão acima da velocidade máxima permitida. Mas, será correto que o motorista do carro a 82 Km/h seja multado?

Para uma melhor classificação em situações como esta é que são utilizadas as funções de *membership*, que possuem variação suave. Estas funções indicam o grau de pertinência de um elemento em relação a um determinado grupo. No exemplo acima, o carro com 87 Km/h com certeza possui um grau de *membership* bem maior, em relação ao conjunto de carros acima da velocidade máxima permitida do que o carro com velocidade de 82 Km/h, porém terá um grau de *membership* menor, em relação ao carro com velocidade de 90 Km/h.

Na primeira situação poderia ser utilizada a função de *membership* ilustrada na Figura 4.2, que mostra uma função *membership* de variação brusca. Esta função, de acordo com a variação de velocidade (eixo horizontal), varia o grau de *membership* (eixo vertical). Este valor pode variar entre 0 e 1.

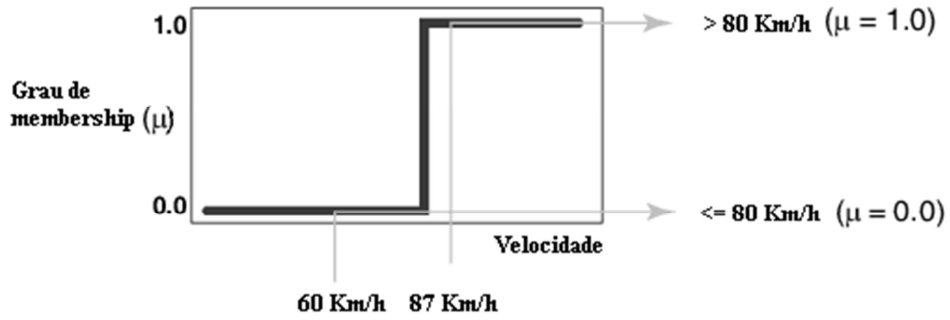

Figura 4.2 – Exemplo de função *membership* brusca.

Na segunda situação, onde têm-se uma mudança no valor de *membership* mais suave, é apresentada a função mostrada na Figura 4.3.

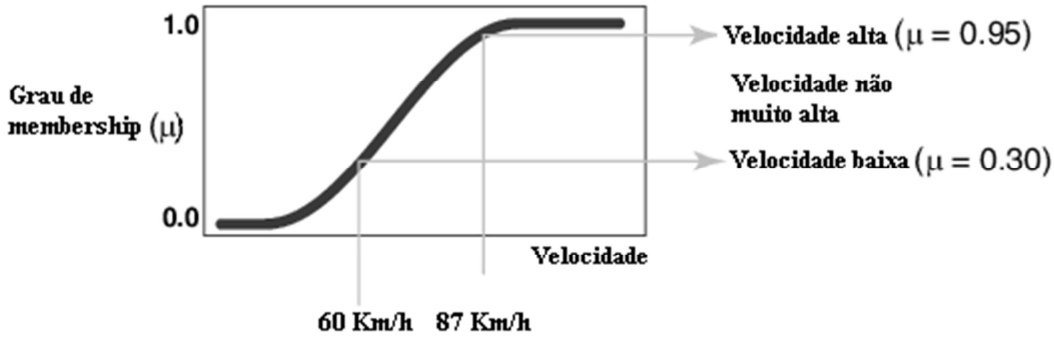

Figura 4.3 – Função *membership* suave.

A função de *membership* suave, onde de acordo com a variação de velocidade (eixo horizontal), o valor do grau de *membership* (eixo vertical) varia de forma suave entre 0 e 1.

Note que na primeira classificação (com variação brusca do *membership*) um carro tem a velocidade considerada alta e o outro com velocidade baixa. Já na segunda classificação (com variação suave do *membership*) um carro é considerado com velocidade baixa e o outro com velocidade não muito alta, ou seja, o segundo carro não é considerado com velocidade alta, mas sim um grau entre velocidades consideradas altas e baixas.

De acordo com a aplicação, o formato destas funções de *membership*s pode variar. Em um sistema real, a função de *membership* pode ser baseada em informações fornecidas por um operador do sistema que conhece a faixa de valores de variação de um determinado evento e a ação a ser tomada para cada situação. Através destas informações pode ser construída uma função de *membership* que mais se aproxima com os valores fornecidos pelo operador do sistema [25].

As Figuras 4.4, 4.5, 4.6, 4.7 e 4.8 apresentam as funções de *membership* mais simples. É possível notar que cada função tem um grau de inclinação diferente, o que as torna mais adequadas em determinadas aplicações específicas.

A função mostrada na Figura 4.4 é um exemplo de uma função *membership* de formato triangular que possui variações bruscas entre o valor máximo 1 e o valor mínimo 0. Para se obter esta função têm-se:

Para $x = [0,3] \rightarrow \mu = 0;$

Para $x =]3,6[\rightarrow \mu = \frac{x-3}{3};$

Para $x = 6 \rightarrow \mu = 1;$

Para $x =]6,8[\rightarrow \mu = \frac{8-x}{2};$

Para $x \geq 8 \rightarrow \mu = 0.$

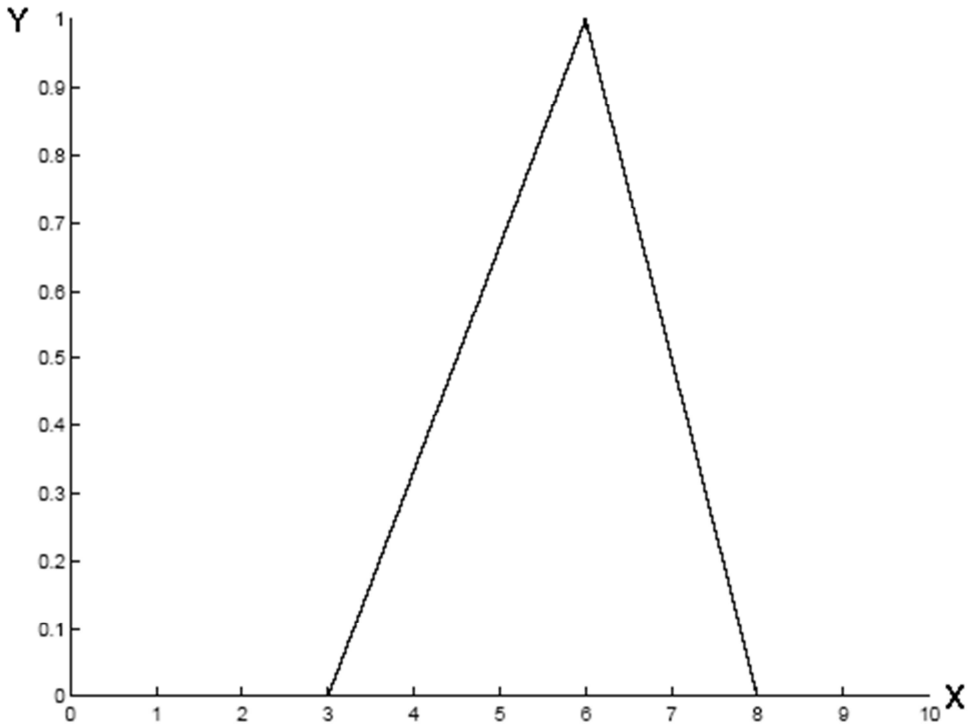

Figura 4.4 – Função *membership* triangular.

A Figura 4.5 mostra uma função *membership* sigma, que possui uma variação mais suave, se comparada com as funções *membership* analisadas anteriormente. Esta função, em um caso especial, pode ser calculada da seguinte maneira:

$$\mu = \frac{1}{1 - e^{-x}} \tag{4.1}$$

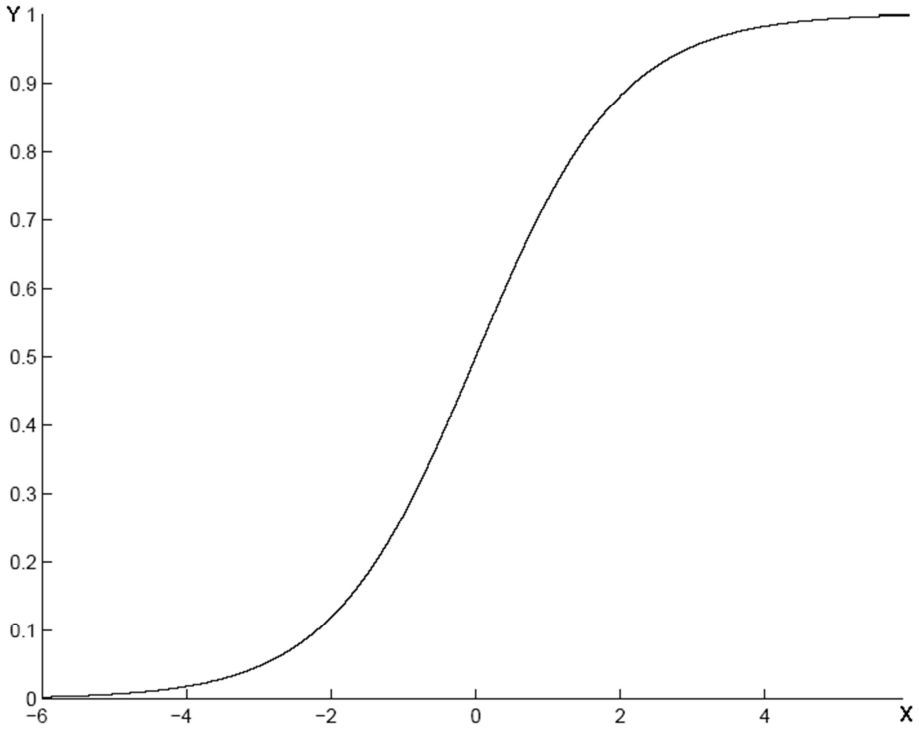

Figura 4.5 – Função *membership* sigma.

Existem também as funções de *memberships* descritas por funções de distribuição Gaussiana normalizadas, como ilustrado na Figura 4.6.

Se comparada com as outras funções anteriores a função *membership* Gaussiana, ilustrada na Figura 4.6, é de variação mais suave e terá um melhor resultado em sistemas que exigem em sua saída funções de variações suaves. Esta função, de acordo com a variação dos valores de x, no eixo horizontal, apresenta em sua saída, no eixo vertical a função:

$$\mu(x, \sigma, \zeta) = e^{-\frac{(x-\zeta)^2}{2.\sigma^2}} \tag{4.2}$$

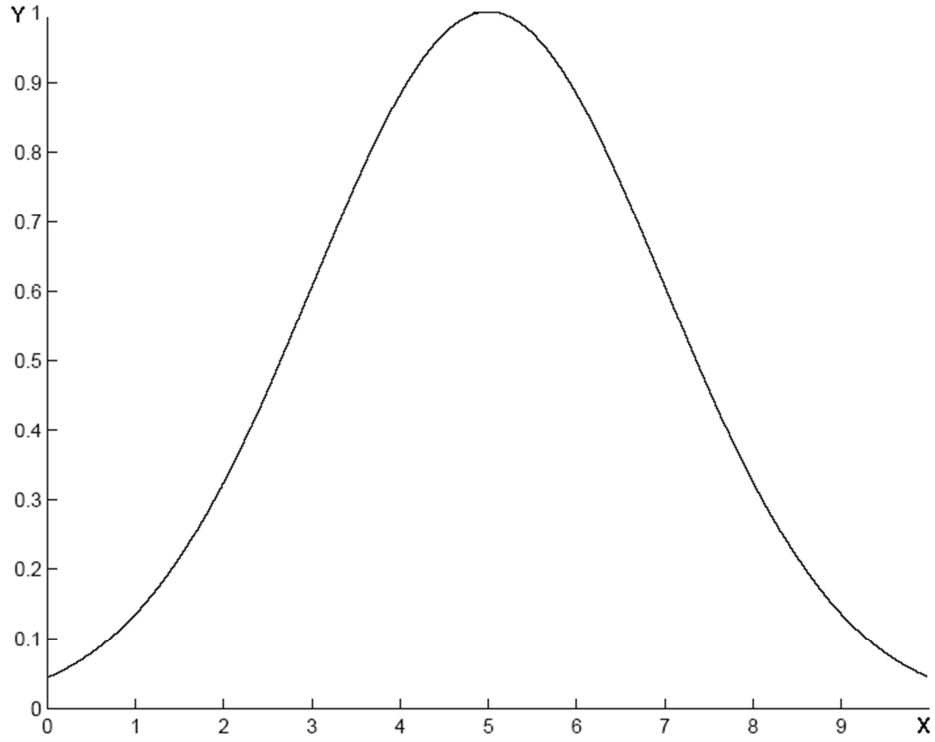

Figura 4.6 – Função *membership* Gaussiana.

A Figura 4.7 mostra um exemplo de uma função que é formada pela diferença de duas funções sigmoidais. Esta função também possui uma variação mais suave e de acordo com a variação de x, no eixo horizontal, apresenta o valor de resposta, no eixo vertical definida pela função:

$$\mu(x,\alpha_1,\zeta_1,\alpha_2,\zeta_2) = [1+e^{-\alpha_1(x-\zeta_1)}]^{-1} - [1+e^{-\alpha_2(x-\zeta_2)}]^{-1} \qquad (4.3)$$

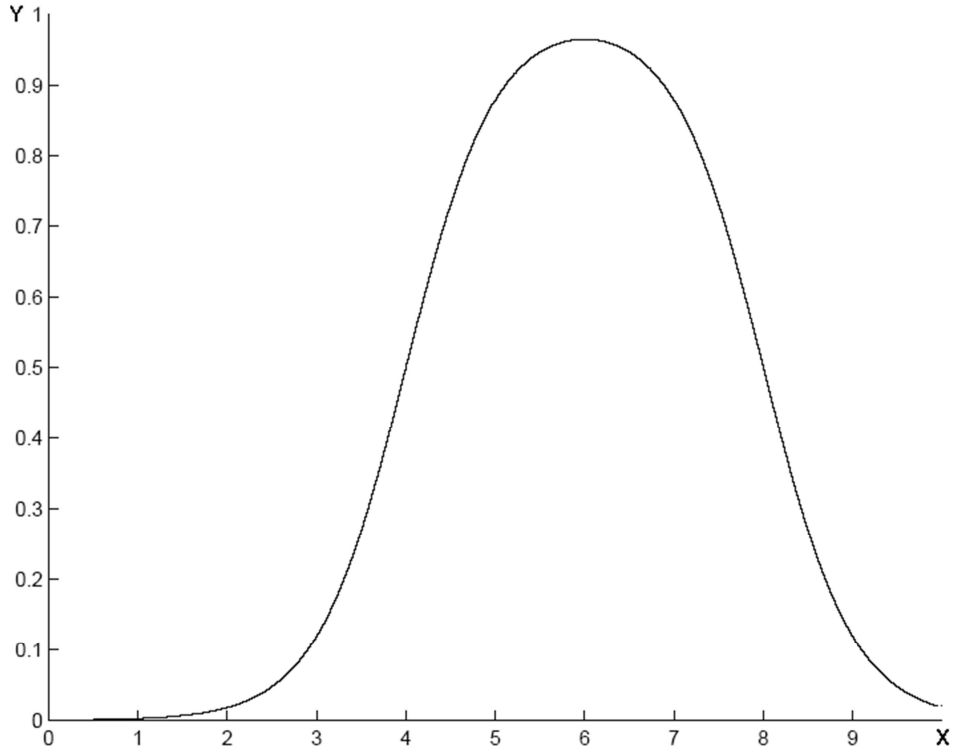

Figura 4.7 – Função *membership* formada pela diferença de duas funções sigma.

A função da Figura 4.8 é conhecida pelo seu formato como função sino. Nota-se uma variação mais brusca entre seu valor máximo e mínimo. Esta função *membership* pode ser definida pela função:

$$\mu(x,\alpha,\beta,\zeta)=\left[1+\left|\frac{x-\zeta}{\alpha}\right|^{2\beta}\right]^{-1} \tag{4.4}$$

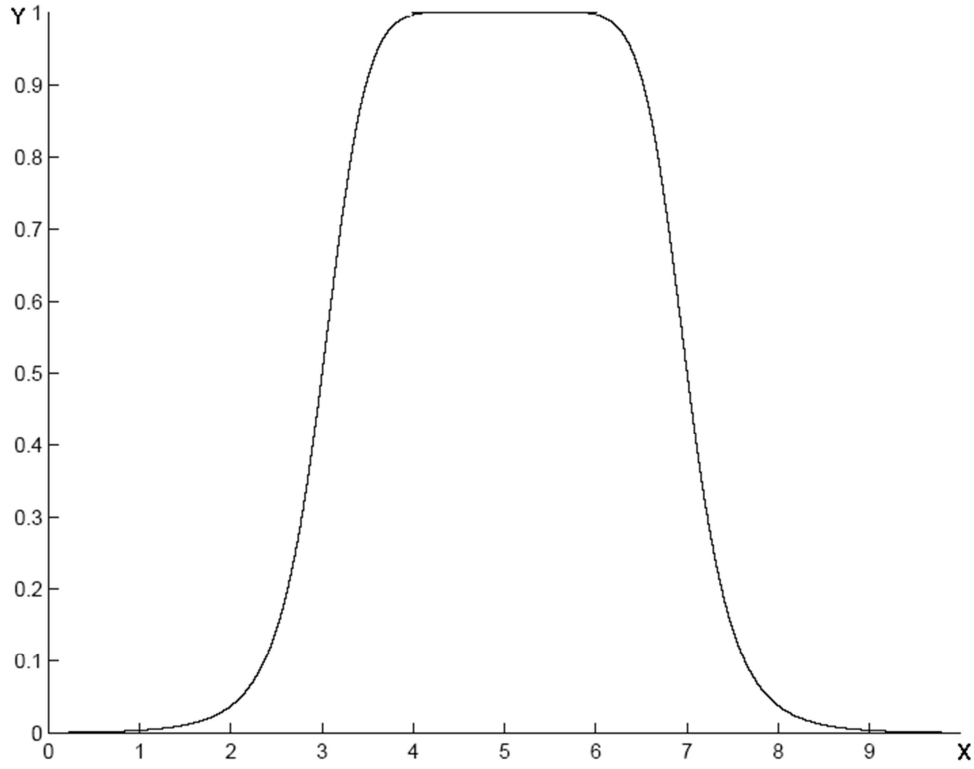

Figura 4.8 – Função *membership* sino generalizada.

De acordo com [25] os formatos das funções de *memberships* que normalmente são mais utilizados são o triangular e o trapezoidal, devido à facilidade para gerar estas funções. Porém, em situações onde desempenho suave é considerado de grande importância, as funções de *membership* podem ter outros formatos como gaussiana, sigmóide e sino.

Para verificar e demonstrar a influência do formato das funções *membership*, no funcionamento de um sistema de *Fuzzy Clustering*, serão analisadas três funções de formatos diferentes atuando em uma mesma aplicação. Nesta análise todas as funções de *memberships* e todo o funcionamento do sistema foi feito em MatLab.

O sistema utilizado para teste utiliza as funções de *membership* para criar regras de controle. Neste exemplo, a função de controle deverá ser sen($x_1.x_2$), ou seja, para valores de X_1 e X_2 de entrada, a saída deve gerar o resultado da função de controle.

Inicialmente são fornecidos, para o treinamento do programa, alguns valores de entrada do sistema e os valores de saída que devem ser gerados. Após o treinamento com os valores de dados fornecidos, o programa gera uma função de controle que será capaz de gerar em sua saída valores de acordo com o treinamento feito anteriormente. Esta aplicação deverá utilizar 14 funções de *membership* e como primeiro exemplo, estes *membership* t6em o formato sino e devem atuar em toda a faixa de variação possível das entradas

Como exemplo, o programa descrito acima terá que gerar uma função em sua saída que gere o valor de sen(x_1.x_2), de acordo com dois valores de entrada x_1 e x_2 . O gráfico da Figura 4.9 mostra a função correta esperada na saída do programa onde se tem a variação de x1 e x2 entre -3 e 3 e o resultado de sen(x1.x2), que varia entre -1 e 1.

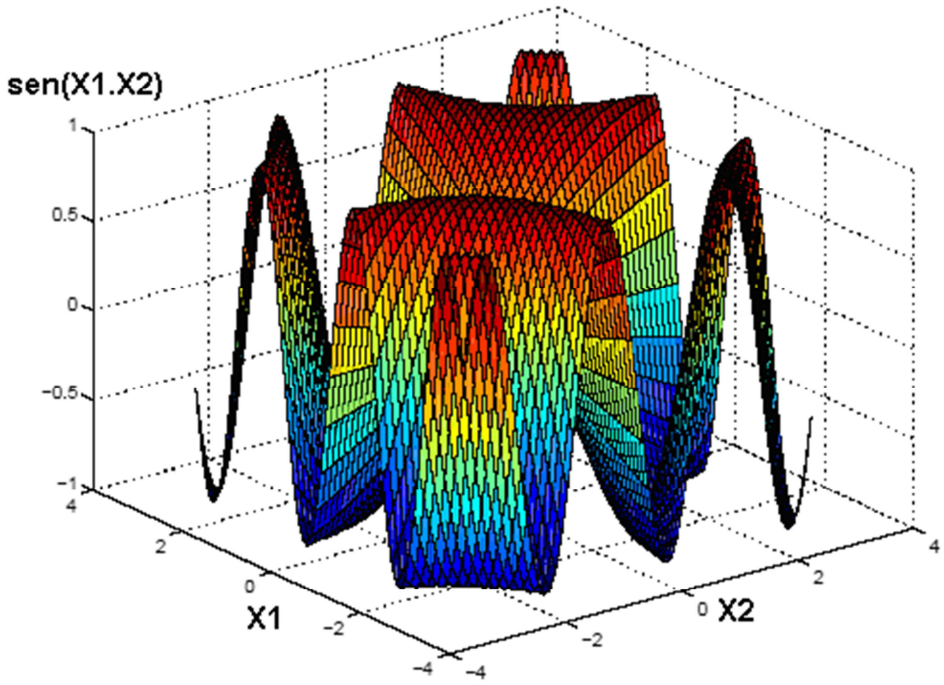

Figura 4.9 – Função sen(x_1.x_2).

Na Figura 4.10 é apresentado o gráfico das funções *membership* sino que será usada para verificar a viabilidade deste sistema.

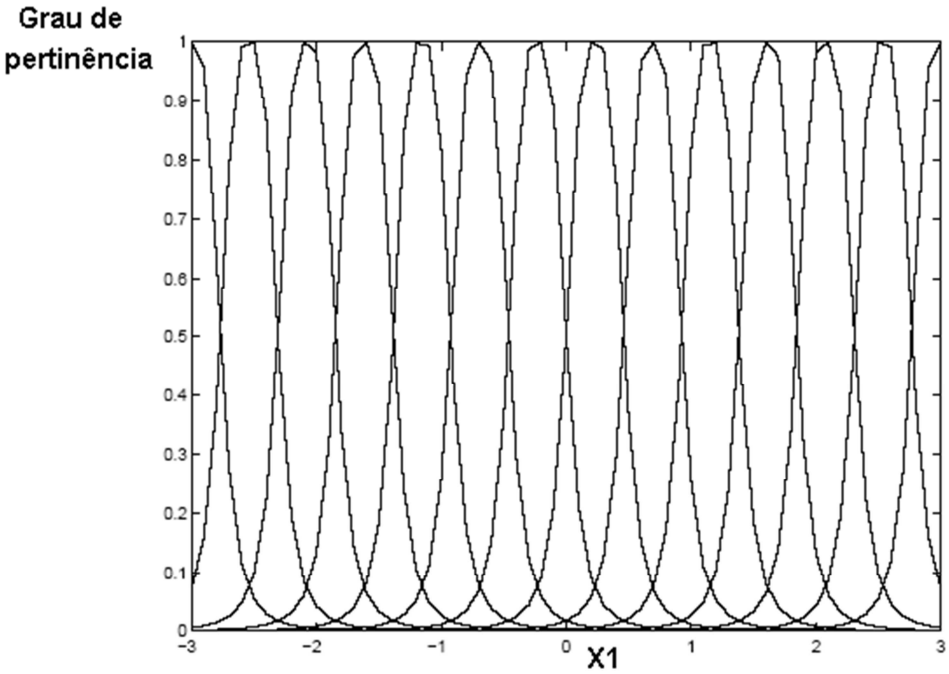

Figura 4.10 – Gráfico *membership* sino.

Este é o gráfico dos 14 *memberships* utilizados pelos dois dados de entrada (x1 e x2). Como o programa está gerando uma função que relaciona x1 e x2, se cada entrada trabalha com 14 *memberships*, o programa irá relacionar os 14 *memberships* de cada entrada, resultando em uma combinação de 196 *memberships*.

O gráfico da Figura 4.11 mostra a função gerada pelo programa utilizando os 14 *memberships* para cada entrada. Este gráfico mostra que o sinal gerado na saída do programa obteve alguns erros, que são indicados pelas pequenas ondulações geradas na superfície da figura e que não apareceram no gráfico da função esperada da Figura 4.9.

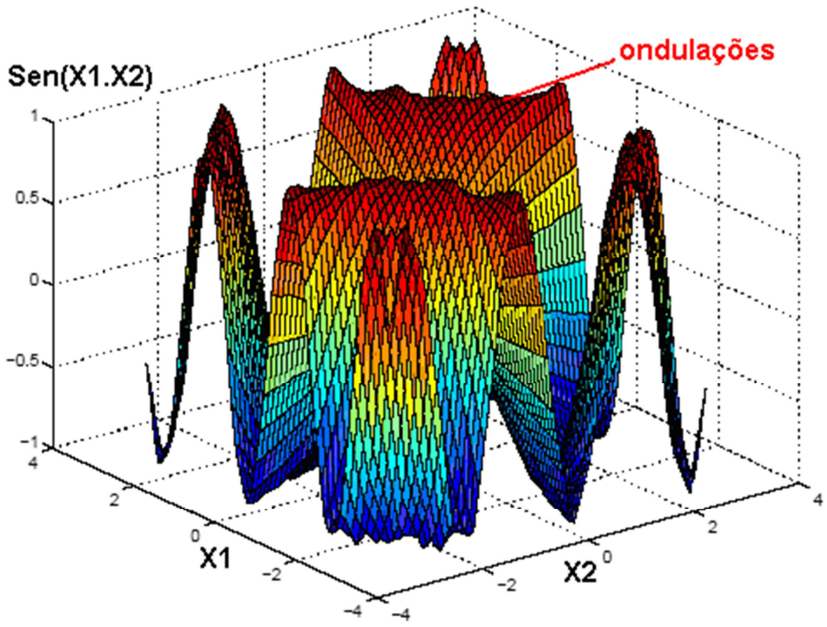

Figura 4.11 – Gráfico gerado pelo programa para *membership* sino.

Na Figura 4.12 é apresentado o gráfico que mostra os erros gerados pelo programa.

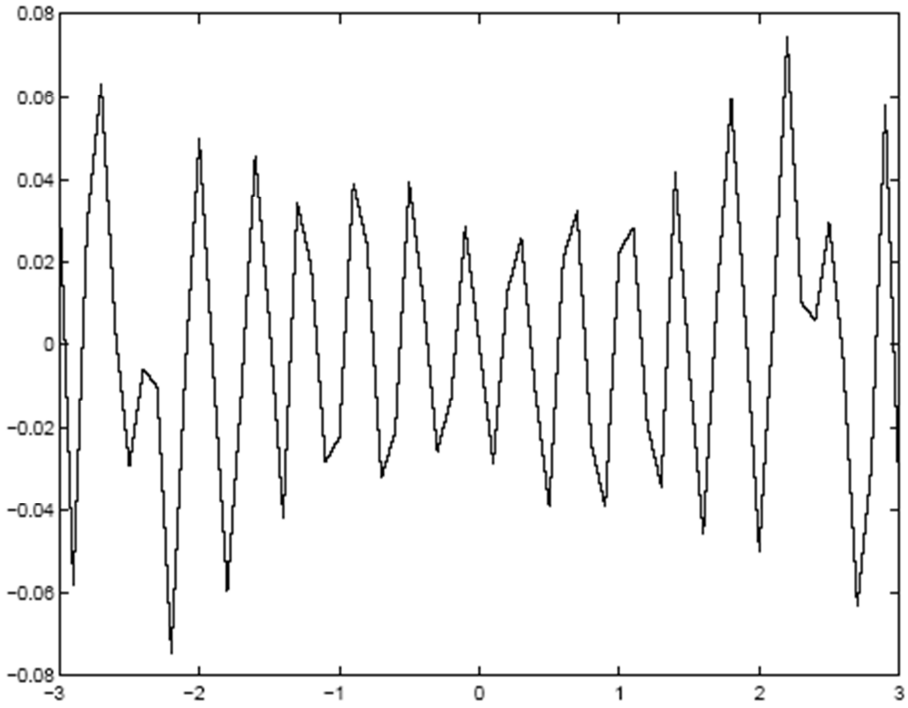

Figura 4.12 – Gráfico do erro gerado entre o sinal esperado para o sistema e o sinal gerado do sistema.

O gráfico do erro gerado pelo programa foi calculado subtraindo a função gerada pelo programa da Figura 4.11, pela função esperada da Figura 4.9. Verifica-se que dentro do intervalo entre -3 e 3, o maior erro gerado foi próximo de 0,08.

Agora será utilizado a mesma função do exemplo anterior, sen($x_1.x_2$), mas a função de *membership* terá outro formato. Neste exemplo tem-se uma função *membership* de Gauss. Na Figura 4.13 é apresentada a função de saída sen($x_1.x_2$) que se espera do programa. De acordo com os valores de x1 e x2, que neste caso podem variar entre -3 a 3, e utilizando a função de *membership* de Gauss, é gerada uma função sen($x_1.x_2$), que poderá variar entre -1 a 1.

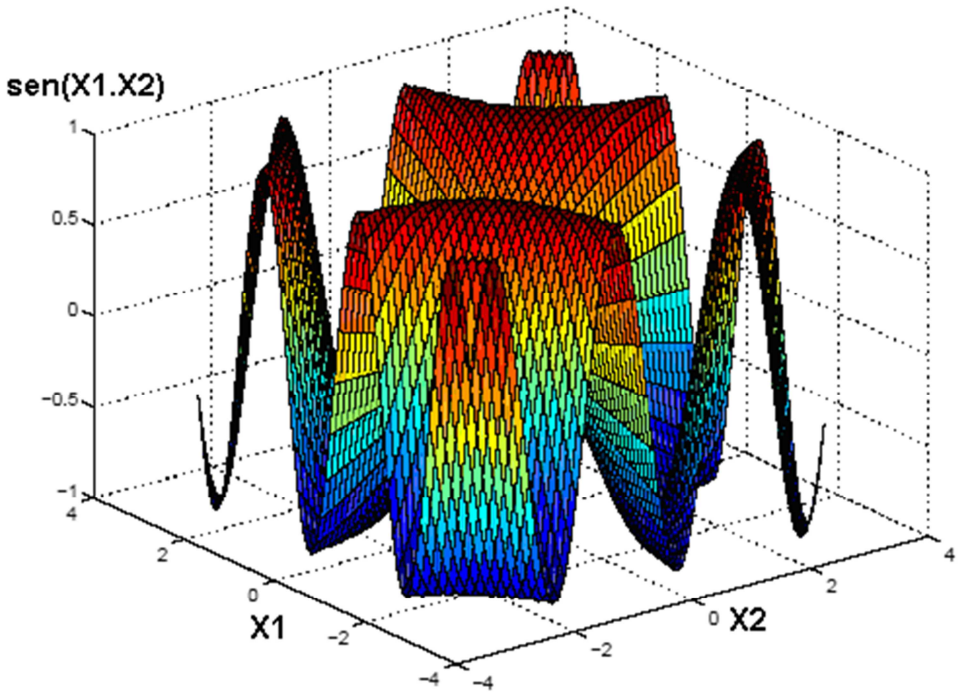

Figura 4.13 – Gráfico da função sen($x_1.x_2$).

De acordo com a Figura 4.14, que apresenta o gráfico da função *membership* de Gauss, nota-se que a variação das funções de *memberships* é suave se comparado com o gráfico de *membership* de formato sino. Assim para gerar uma função seno que possui uma variação suave ela apresenta um melhor resultado, ou seja, um erro menor.

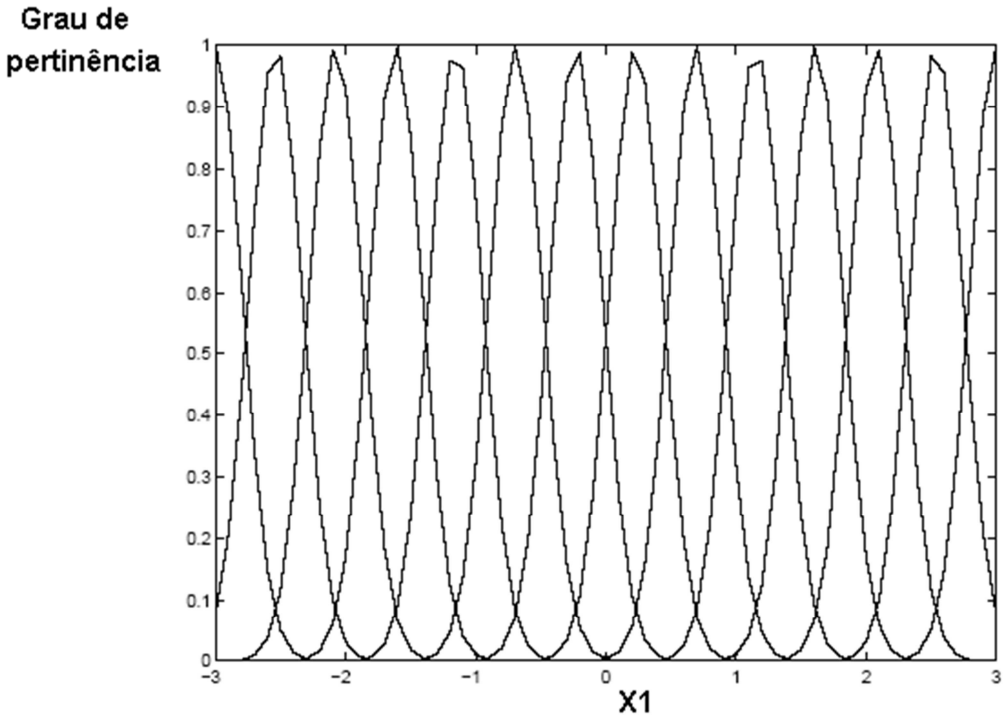

Figura 4.14 – Gráfico *membership* Gauss.

A Figura 4.15 mostra a função de resposta gerada pelo programa que recebeu em sua entrada x_1 e x_2, variando entre -3 a 3 e gerou em sua saída uma função que deveria se aproximar da função sen(x_1.x_2), utilizando o *membership* de Gauss. Comparando o gráfico de saída para este *membership*, nota-se que praticamente não ocorreram ondulações na superfície da figura gerada.

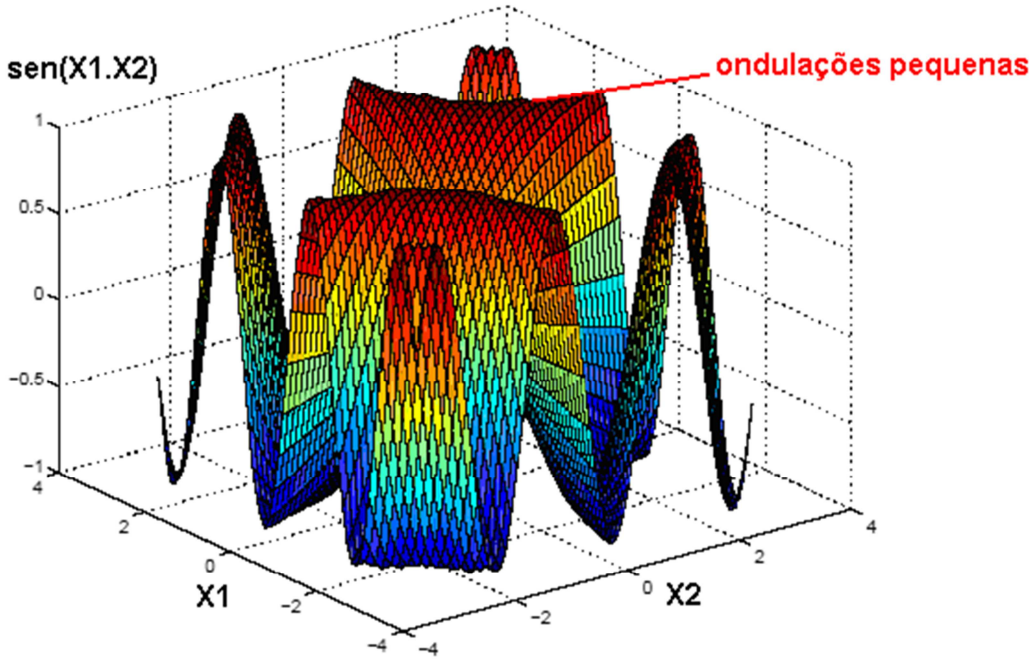

Figura 4.15 – Gráfico de saída do programa para *membership* Gauss.

De acordo com a Figura 4.16 o maior erro gerado entre a função esperada, dentro do intervalo entre -3 a 3, e a função gerada foi próximo de 0,03. Comparando com o erro gerado pela função que utilizou o *membership* sino, o erro foi menor.

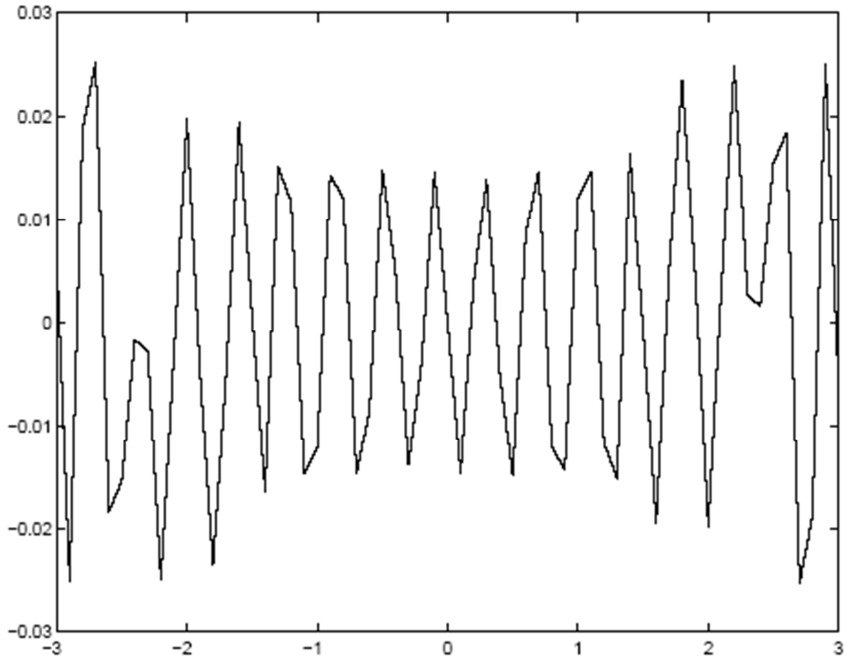

Figura 4.16 – Gráfico do erro gerado entre a função sen(x_1.x_2) e a função gerada pelo programa, utilizando *membership* Gauss.

O próximo teste irá utilizar a mesma função, que com a variação das entradas x_1 e x_2 entre -3 a 3, deve gerar em sua saída a sen(x_1.x_2) mostrada na Figura 4.17. Neste caso serão utilizadas 14 funções de *membership* onde, destas funções, 10 tem o formato triangular e 2 tem formato trapezoidal.

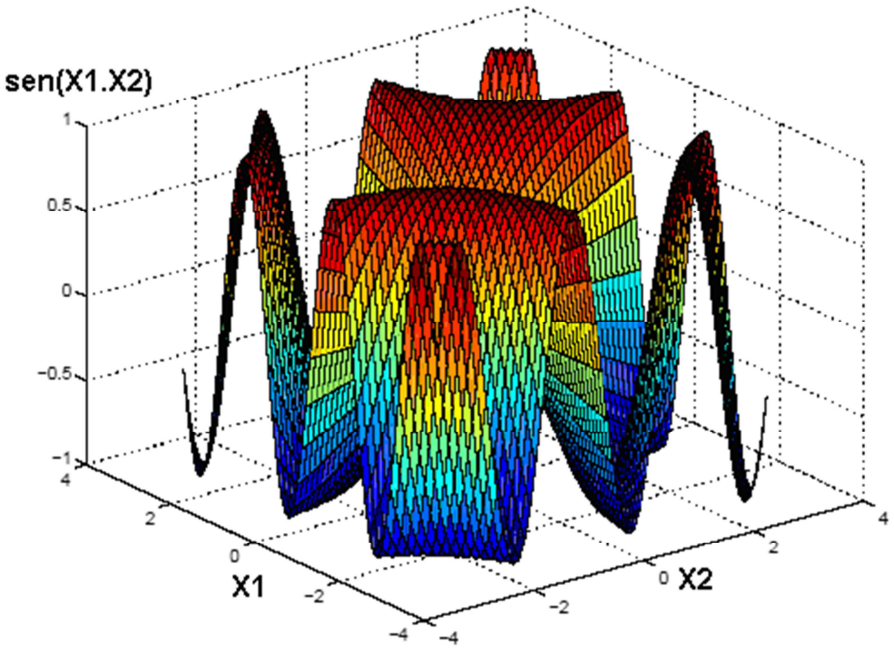

Figura 4.17 – Gráfico da função sen(x_1.x_2) a ser gerada pelo programa.

A Figura 4.18 mostra as 14 funções de *membership* utilizadas para cada entrada (x_1 e x_2). Como estas duas entradas serão associadas para gerar a função de saída sen(x_1.x_2), este programa terá um total de 196 regras.

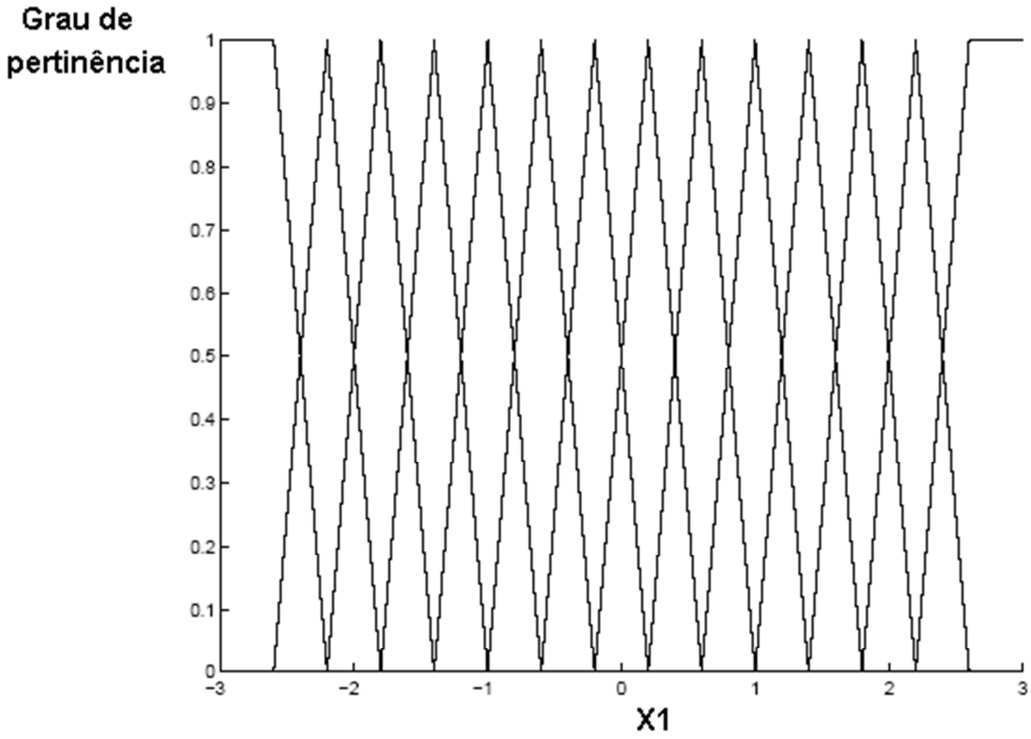

Figura 4.18 – Gráfico *membership* triangular.

A Figura 4.19 apresenta o gráfico gerado pelo programa, onde para x_1 e x_2 variando entre -3 a 3, o programa gerou uma função em sua saída que se aproxima da função esperada mostrada na Figura 4.17. Observa-se que o gráfico gerado pelo programa está bastante deformado em relação aos gráficos gerados pelas outras funções de *memberships*, Sino e Gauss. Esta deformação é percebida pelas ondulações na superfície da figura indicada. Isto ocorre porque como a função a ser gerada pelo programa é uma função de variação suave e a função de *membership* de formato triangular possui uma variação mais brusca. Se comparada com as outras duas funções, a função triangular foi a que apresentou maior deformação na função de saída. A função de Gauss, que é a função de *membership* mais suave, foi a que obteve melhor resultado.

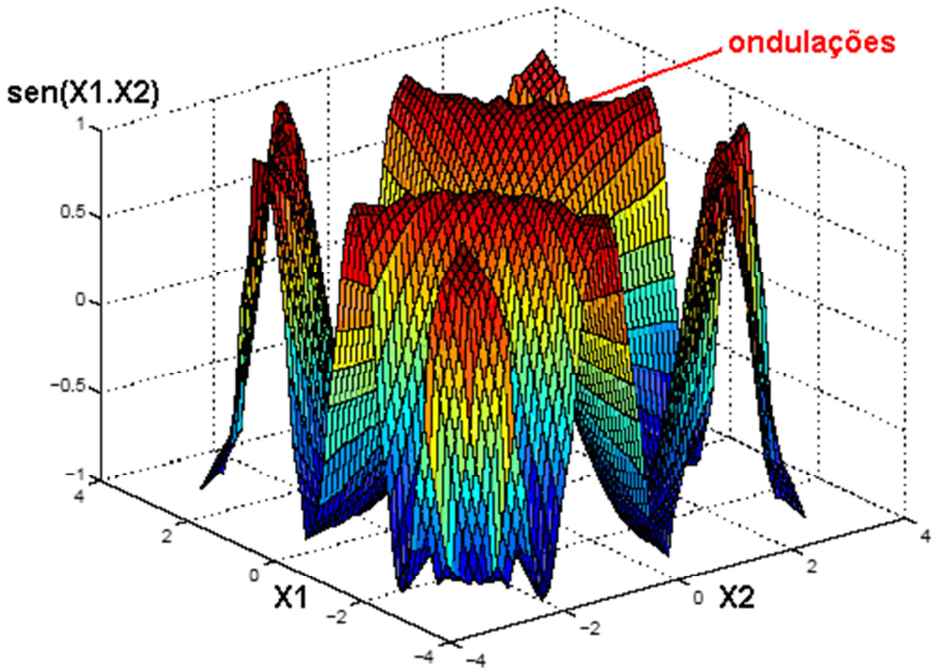

Figura 4.19 – Gráfico de saída para *membership* triangular.

A Figura 4.20 mostra a função do erro gerado pelo programa. Este erro foi gerado pela subtração da função gerada pelo programa na figura 4.19 pela função esperada da figura 4.17. Nota-se que o maior erro chegou próximo de 0,6. O erro que utilizou a função de *membership* triangular foi maior, devido ao fato de que esta função possui variações mais bruscas, não apresentando um bom resultado para a função sen(x_1.x_2).

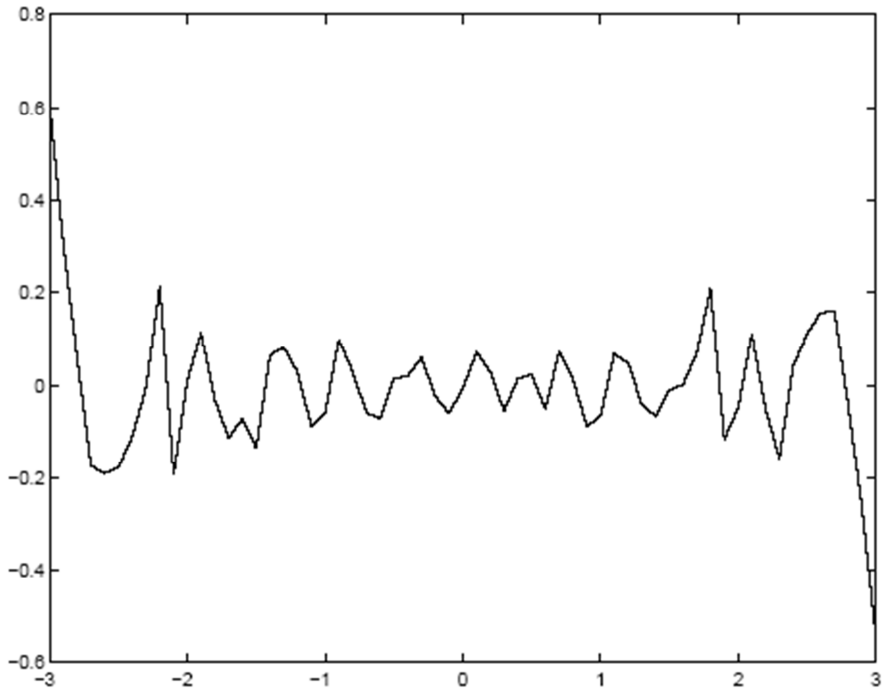

Figura 4.20 – Gráfico do erro gerado pelo programa entre a função desejada e a função a ser gerada.

De acordo com os resultados obtidos nestes exemplos apresentados anteriormente, utilizaremos o formato de *membership* que proporcionou um melhor resultado, no caso a função de Gauss.

Existe ainda uma característica que também irá influenciar na precisão de resposta do sistema que é a quantidade de conjuntos de *memberships*. Quanto maior a quantidade de *memberships* melhor será a resposta do sistema, porém também será maior a demanda computacional necessária para este sistema. Segundo [25] uma quantidade entre 2 a 7 *memberships* é o suficiente pois um aumento de 5 para 7 conjuntos de *memberships* triangulares resulta em uma melhora de 15%. Um número maior do que 7 *memberships* não mais resulta em melhoras significativas no sistema.

4.3. Criação de Regras

Após a escolha da função de *membership* que gerou o melhor desempenho do sistema, será descrito o processo matemático para a geração de regras. Neste processo é que são criados os clusters que descrevem as principais características do sinal.

Inicialmente devem ser coletados vários dados de entrada e as respectivas saídas que estes dados devem provocar no sistema, para o treinamento do algoritmo a ser criado.

Dado uma sequência de valores de entrada de um sistema X, onde X = [x_1,x_2,x_3,....,x_k], com k representando a quantidade de entradas do sistema para o treinamento e Y, onde Y = [y_1, y_2, y_3, ..., y_k], que são as respectivas saídas do sistema quando nele são aplicadas as entradas X. Com uma quantidade de regras M pode-se definir a quantidade de *memberships* a serem gerados, que deve ser igual ao produto da quantidade de *memberships* de cada evento [26].

Cada evento representa uma variável de um sistema a ser controlado. Como exemplo, considere um sistema de controle de um robô que deve colocar um objeto em um determinado local. Para que o sistema consiga fazer o robô colocar o objeto corretamente no local, este sistema deve ter a informação, através de sensores, da altura deste objeto, da distância até o local onde o objeto deve ser colocado, altura deste local e outros. Cada informação desta seria um evento a ser analisado pelo sistema [26].

Definida a quantidade de regras a serem utilizadas, será utilizado um método de estimação pelos mínimos quadrados de um dado aplicado, para determinar os coeficientes a_i e b_i da equação de cada regra. De acordo com o modelo *Fuzzy* [25], se uma entrada x pertence a um *membership* logo tem-se A_i, onde A_i representa o grupo de antecedente *fuzzy* de cada regra, e assim, a saída correspondente a uma determinada regra i será [26]:

$$y_i = a_i^T x + b_i \tag{4.5}$$

A quantidade de coeficientes da equação de saída vai depender da quantidade de eventos que serão analisados na entrada do sistema. Sendo n o número de eventos de entrada do sistema, deve-se calcular sempre (n + 1) coeficientes da equação de saída de cada regra [26].

Para uma determinada entrada *x(k)* o valor total de saída do sistema *y(k)* para esta entrada será calculado através da equação 4.6:

$$y(k) = \sum_{i=1}^{M} u_{ki} y_i(k) \tag{4.6}$$

Onde u_{ki} é o grau normalizado de participação de uma entrada para a regra Ri e é calculado pela equação 4.7.

$$u_{ki} = \frac{A_i(x_k)}{\sum_{i=1}^{M} A_i(k)} \tag{4.7}$$

Para determinar os parâmetros de cada regra considera-se uma coleção de N pares de dados de entradas e saídas de um sistema {x(k), y(k)} onde k = 1, 2, 3, ..., N, com x(k) sendo um vetor que contém todas as entradas [x1(k), x2(k), ..., xn(k)] e y(k) o vetor das respectivas saídas correspondentes [y(1), y(2), ..., y(k)]T. Monta-se uma matriz Xe, que é uma matriz com todos os dados de entrada em uma coluna e uma coluna unitária [X, 1] com colunas [x(k)T, 1]. A ativação de cada regra é feita pela matriz γi, que é uma matriz diagonal onde os elementos da diagonal principal são retirados da coluna i do grau normalizado das entradas para a regra i, que são denotadas por u_{ki}. Monta-se agora a matriz X' que é uma matriz de dimensões N x M.N, composta pela multiplicação das matrizes γi e Xe, como mostrado abaixo [26] pela equação 4.8

$$X' = [\gamma_1.Xe, \gamma_2.Xe, ..., \gamma_M.Xe] \tag{4.8}$$

Com a montagem das matrizes anteriores pode-se calcular o vetor θ', que é um vetor de dimensões M.(n + 1) x 1 e que apresenta os coeficientes finais das regras a serem geradas.

$$\theta' = [\theta_1^T, \theta_2^T, ..., \theta_M^T]^T \tag{4.9}$$

Cada elemento do vetor θ' apresenta os coeficientes de cada regra,

$$\theta_i^T = [a_i, b_i] \tag{4.10}$$

Logo o modelo de regressão pode ser obtido pela expressão 4.11:

$$y = X'.\theta' + e \tag{4.11}$$

Nesta equação e é o erro de aproximação. Para o cálculo de θ' utilizando o modelo de regressão, têm-se:

$$\theta' = \left[(X')^T . X' \right]^{-1} . (X')^T . y \tag{4.12}$$

4.4. O Processo *Fuzzy Clustering*

Um processo de *Fuzzy Clustering* consiste em dividir um determinado grupo de dados em grupos ou clusters, de acordo com a distância dos pontos destes dados em relação ao protótipo de clusters. Existem vários métodos de identificação destes clusters e nesta aplicação foi proposto o método GK *Fuzzy Clustering* algorithm [27]. Todo cluster representa uma regra dentro de um grupo de regras.

De posse dos pares de valores das entradas e suas saídas correspondentes, é criada uma matriz de regressão X e o vetor de saída y, como mostrado abaixo:

$$X^T = [x_1, x_2, ..., x_N] \quad y^T = [y_1, y_2, ...y_N] \tag{4.13}$$

Onde N indica o número de pares utilizados para a identificação. Lembrando que, o número de entradas utilizadas deve ser maior que o número de regras n, a serem utilizadas.

Os grupos de antecedentes fuzzy são determinados através de medidas do produto do espaço para os sistemas de entrada e saída. O grupo de dados $Z \in R^{(1+n)\times N}$ é representado por uma matriz formada por X e y:

.

$$Z^T = [X, y] \tag{4.14}$$

Cada coluna de Z contém um par de dados de entrada e saída.

Tendo Z e um número estimado M de regras, a repartição *Fuzzy* é representada pela matriz *U NxM*, onde cada elemento u_{ki} pertence ao intervalo [0,1] representa o grau de *membership* para o elemento Z_k no cluster i.

Como exemplo de aplicação deste processo de *Fuzzy Clustering*, será apresentado uma simulação, em MatLab, que deverá gerar em sua saída uma função sen(x_1.x_2). Este simulação contempla 14 funções de *memberships* com o formato da função de Gauss com 75% de superposição.

A Figura 4.21 apresenta a função esperada.

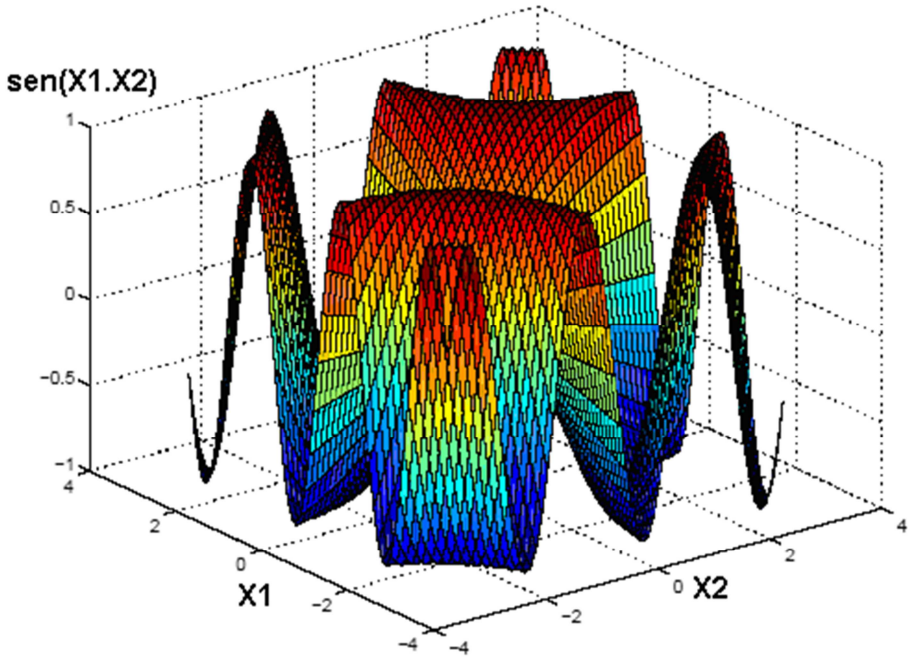

Figura 4.21 – Função que deve ser gerada pelo programa.

Nota-se que é uma função onde as duas variáveis de entrada x1 e x2 variam entre -3 a 3 e a função a ser gerada sen(x_1.x_2) varia entre -1 a 1.

Para gerar a função mostrada acima, utilizam-se funções de *membership* que irão varrer todo o intervalo entre -3 a 3, como mostrado na Figura 4.22.

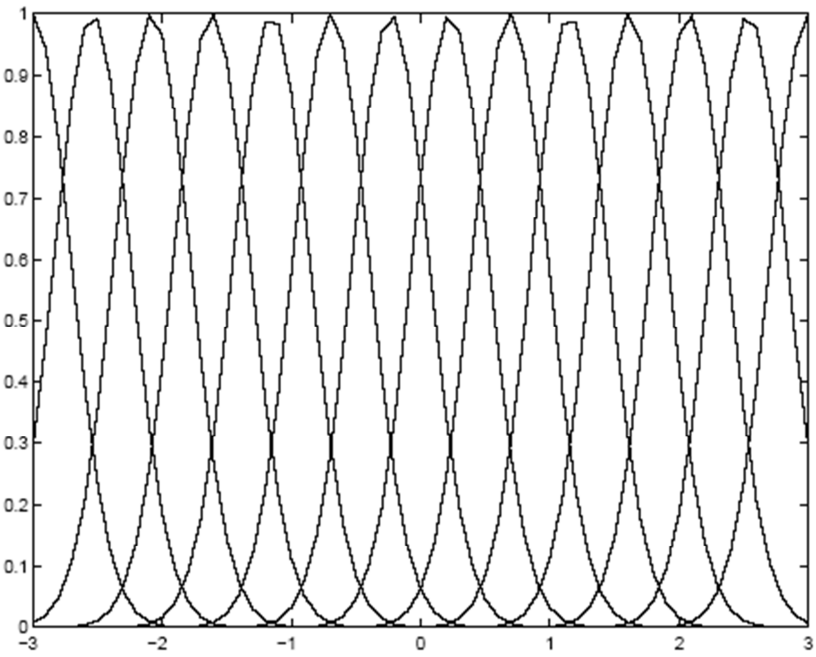

Figura 4.22 – Funções de *memberships* utilizadas pelo programa.

Para este teste foram utilizadas 14 funções de *membership* para cada variável. Estas funções possuem o formato da função de Gauss com uma superposição de funções de aproximadamente 75%.

A Figura 4.23 apresenta a função gerada.

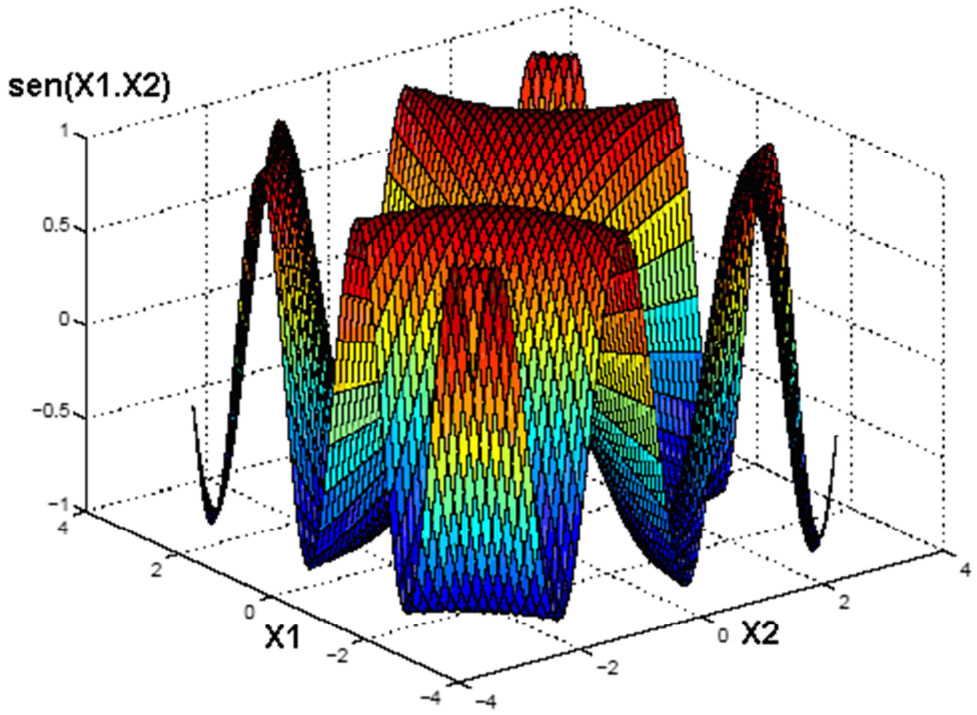

Figura 4.23 – Função gerada na saída do programa.

É possível observar que tendo x_1 e x_2 na entrada do sistema, na saída tem-se uma função que se aproxima da função sen(x_1.x_2), que varia entre [-1,1].

Observa-se que comparando as funções geradas e esperada nota-se que as ondulações que caracterizam um erro da função gerada são quase imperceptíveis. Este pequeno erro pode ser observado na Figura 4.24 com o maior pico de aproximadamente 0.005.

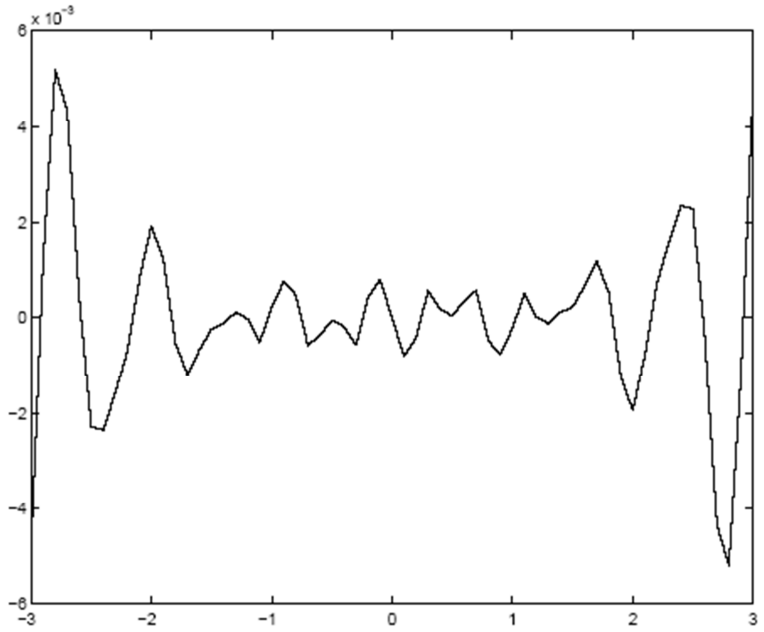

Figura 4.24 – Gráfico do erro gerado pelo programa entre a função esperada e a função gerada.

Nota-se que há erro gerado pelo em toda a faixa de variação dos dados de entrada. Este gráfico foi gerado de tal forma que a variação no eixo vertical indica os valores da subtração da função gerada pelo programa da Figura 4.23 pela função esperada da Figura 4.21.

De acordo com o processo de *Fuzzy Clustering* descrito anteriormente, será mostrado um gráfico com os clusters ou a posição correspondente onde cada regra está atuando, conforme Figura 4.25.

Na Figura 4.25 observa-se que cada ponto azul do gráfico representa a posição de um cluster. Como neste exemplo foram utilizadas 14 funções de *membership* para cada entrada x_1 e x_2, o programa gerou 196 regras ou clusters que terão um peso maior ou menor, variando entre -1 a 1, de acordo com a distância entre o ponto analisado e a posição do cluster.

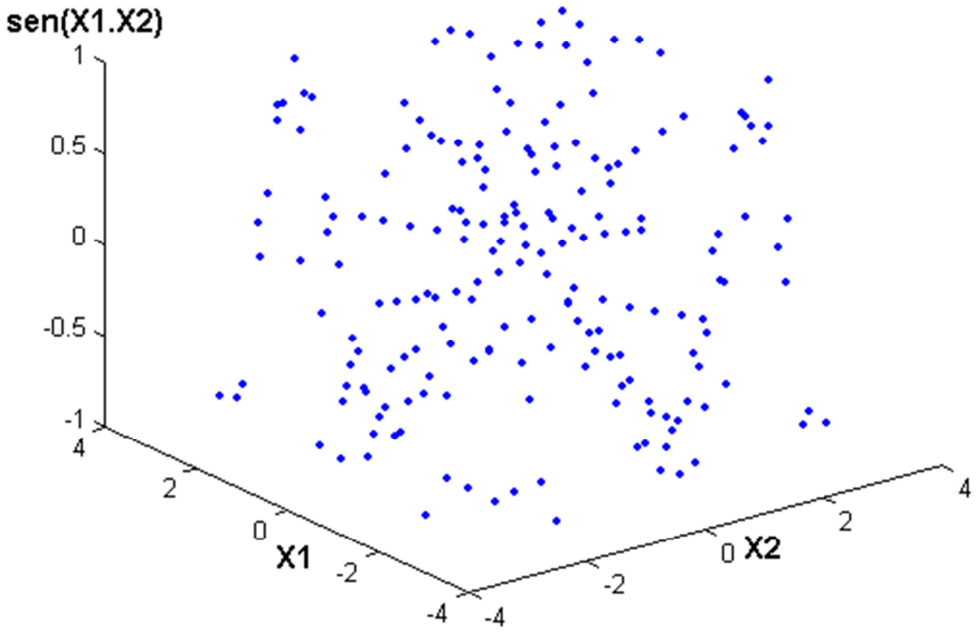

Figura 4.25 – Gráfico que mostra a posição de cada cluster criado pelo programa.

De acordo com os pontos utilizados como teste pelo programa (3721 pontos entre -3 a 3) e os clusters criados pelo programa (14 x 14 = 196 clusters), a Figura 4.26 mostra os clusters e os respectivos pontos de saída do programa.

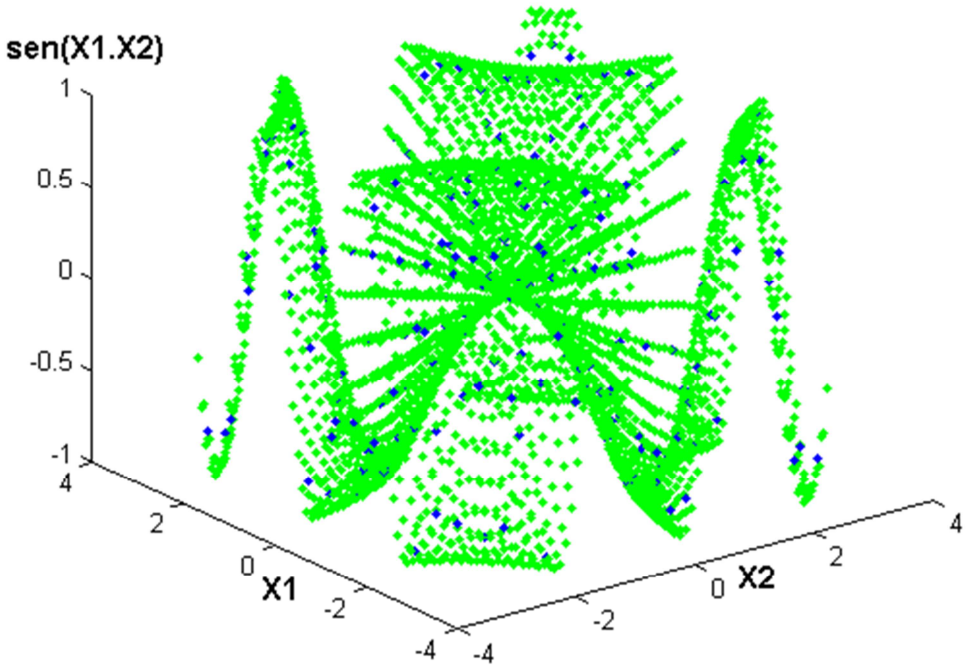

Figura 4.26 – Gráfico que mostra a posição de cada cluster e a posição de cada ponto de saída do programa.

Nota-se a posição dos clusters criados pelo programa, que estão representados pelos pontos azuis, e também a posição de cada ponto gerado pelo programa de acordo com as entradas x_1 e x_2 aplicadas, representados pela cor verde. Nota-se que os clusters gerados estão todos dentro da superfície da função esperada da Figura 4.21.

4.5. Aplicações utilizando o Método *Fuzzy Clustering*

A proposta deste trabalho é utilizar o processo de *Fuzzy Clustering* para criar as regras que são os pontos que descrevem as principais características de variação do sinal de eletrocardiograma, para que a análise de diagnóstico seja feita apenas nestes pontos principais. Como exemplo, será apresentado um sinal de uma derivação de eletrocardiograma e para este sinal será aplicado o processo de *Fuzzy Clustering*.

A Figura 4.27 mostra um exemplo de um sinal de eletrocardiograma com 213 amostras, com tempo de amostragem de 0,003s.

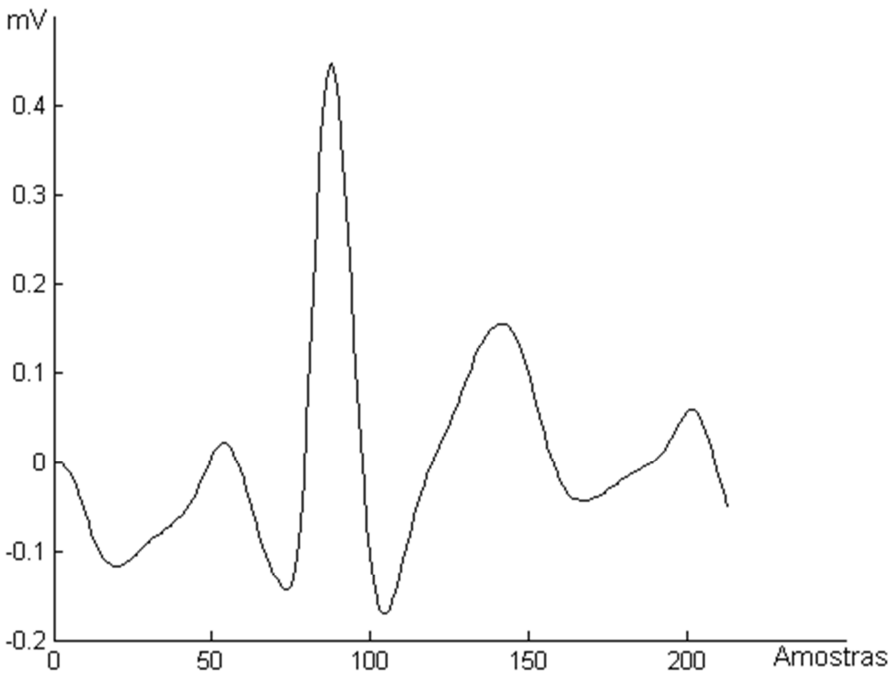

Figura 4.27 – Sinal de eletrocardiograma após pré-processamento.

Para o processo de *Fuzzy Clustering* do sinal de eletrocardiograma da Figura 4.28, foram criadas 20 regras ou pontos que se localizam nas partes mais importantes que caracterizam a variação do sinal. Estas 20 regras foram obtidas depois de diversos testes e

verificações em ambiente de simulação, visto que em uma quantidade menor de regras o erro é muito grande e em uma quantidade maior praticamente não há alterações. Portanto Verificou-se, pelo MatLab, que com as 20 regras consegue-se obter o sinal com praticamente a mesma representatividade qualitativa, obtendo um ganho em termos de tempo de processamento para o hardware, que será mais detalhado no Capítulo 7. A Figura 4.28 mostra a posição de cada ponto característico do sinal, representados no gráfico pelos pontos azuis.

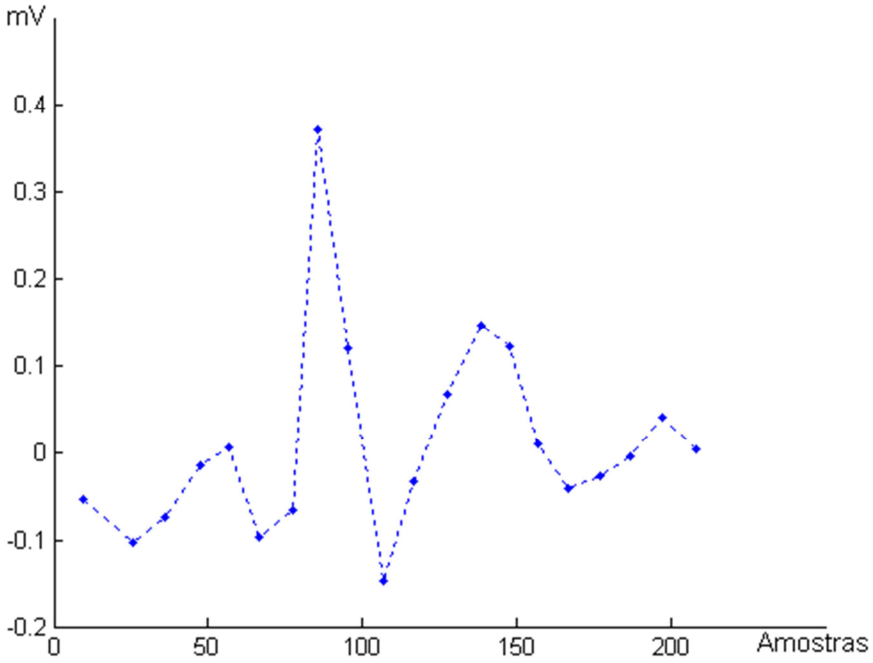

Figura 4.28 – Clusters gerados pelo processo de *Fuzzy Clustering.*

As grandes vantagens do *Fuzzy Clustering*, para aplicações embarcadas e outras possíveis ferramentas de uso desta técnica, são detalhadas no Capítulo 7.

Capítulo 5

Correlação

5.1. Considerações iniciais

Quando a variação de um fenômeno interfere na variação de outro fenômeno, diz-se que existe uma correlação entre eles. Como exemplo, se quando há um aumento em um fenômeno, ocorrer um acréscimo no outro fenômeno ou quando há uma diminuição em um fenômeno existir uma diminuição também no outro fenômeno diz-se que existe uma correlação direta entre eles [28].

Quando o aumento em um fenômeno acarreta uma diminuição em outro fenômeno ou quando a diminuição de um fenômeno acarreta no aumento em outro fenômeno, diz-se que existe uma correlação inversa entre estes dois fenômenos [28].

Neste trabalho será utilizada a correlação entre sinais para se comparar os sinais do eletrocardiograma em análise com os sinais de um banco de dados conhecido.

Visto que um sinal de eletrocardiograma não possui um equacionamento exato, são utilizadas amostras do sinal de eletrocardiograma, pois para este tipo de aplicação é o suficiente, para que seja possível apresentar os possíveis diagnósticos.

5.2. Resultado das Correlações

Quando se analisa a correlação entre dois fenômenos tem-se três possíveis resultados, dependendo do valor do coeficiente de correlação, que nos indica a intensidade da correlação e o tipo de correlação [28].

Quando o coeficiente de correlação é negativo e mais próximo de –1 significa que existe uma forte correlação inversa entre os dois fenômenos analisados. Quando o coeficiente de correlação é negativo ou positivo, porém mais próximo de zero significa que não existe correlação entre os fenômenos analisados. Quando o coeficiente de correlação é positivo e próximo de 1 significa que existe uma forte correlação direta entre os fenômenos em análise [28].

Uma das principais aplicações da correlação no processamento de sinais é encontrar a similaridade entre um sinal desconhecido e um conjunto de sinais conhecidos.

Neste trabalho, de posse dos pontos gerados pelo processo de *Fuzzy Clustering*, foi feito o cálculo da correlação entre estes pontos e os clusters gerados para os sinais de um banco de dados conhecido. Seja X os clusters gerados do sinal em análise e Y os clusters de um sinal do banco de dados de diagnóstico conhecido, assim o índice de correlação entre estes dois sinais é calculado como mostrado na equação [28].

$$\rho = \frac{\sum x \times y}{n \times \sigma x \times \sigma y} \tag{5.1}$$

onde,

$$x = X - MX \tag{5.2}$$

$$y = Y - MY \tag{5.3}$$

MX é a média dos clusters do sinal em análise;

MY é a média dos clusters do sinal do banco de dados;

Assim,

$$MX = \frac{\sum X}{n} \tag{5.4}$$

$$MY = \frac{\sum Y}{n} \tag{5.5}$$

n é o número de clusters;

σx é o desvio padrão de x;

σy é o desvio padrão de y;

$$\sigma x = \sqrt{\frac{\sum x^2}{n}} \tag{5.6}$$

$$\sigma y = \sqrt{\frac{\sum y^2}{n}} \tag{5.7}$$

Como exemplo, sejam os dois sinais mostrados na Figura 5.1. Ao se observar suas características de variação é possível notar que não existe correlação entre eles. Feito o cálculo do índice de correlação entre eles pelo método descrito anteriormente, foi obtido o valor do coeficiente de 0,15, bem próximo de 0. Este coeficiente indica uma correlação fraca, como era de se esperar.

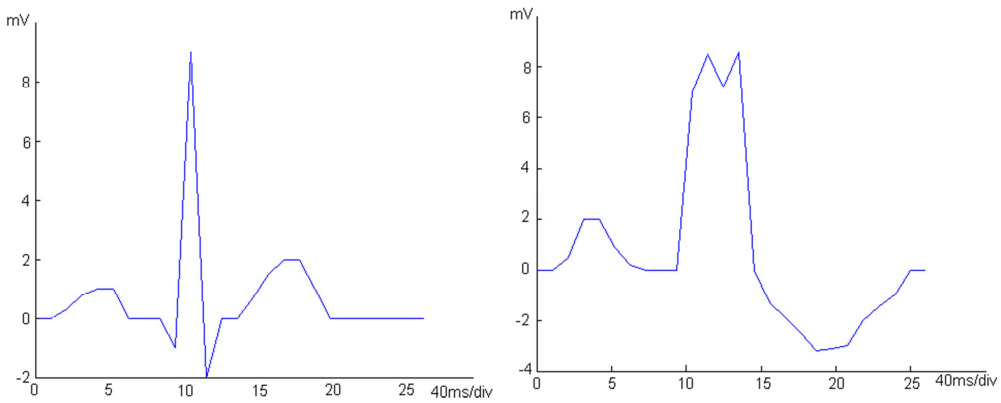

Figura 5.1 – Sinais em análise.

Na Figura 5.2 são apresentados dois sinais onde observa-se que estes possuem uma variação parecida. Foi feito o cálculo da correlação entre eles e o valor obtido do coeficiente foi de 0,89. Este valor indica uma forte correlação direta, como era de se esperar.

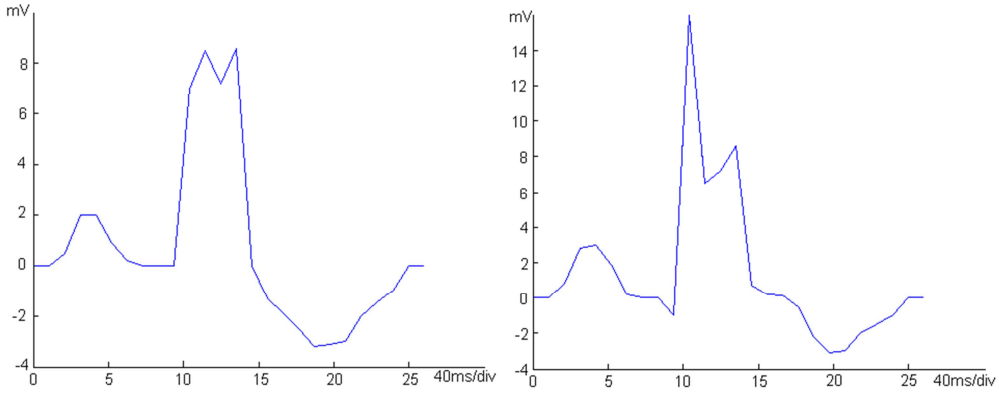

Figura 5.2 – Sinais com forte correlação direta.

Na Figura 5.3 são apresentados dois sinais onde observa-se que estes possuem uma variação inversa. Foi feito o cálculo da correlação entre eles e o valor obtido do coeficiente foi de -0,89. Este valor indica uma forte correlação inversa entre estes sinais, como era de se esperar.

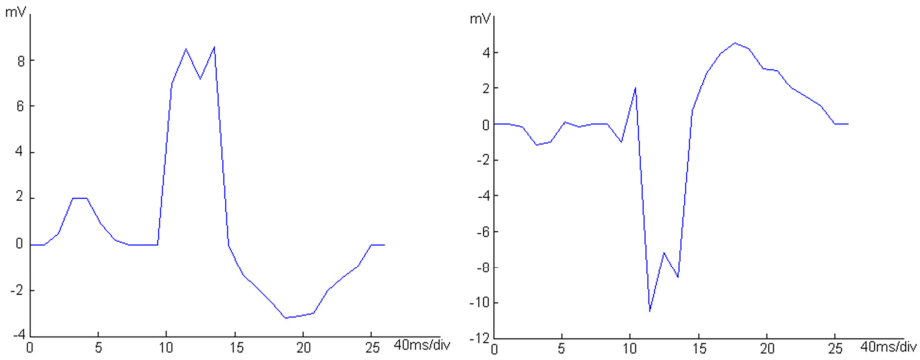

Figura 5.3 – Sinais com forte correlação inversa.

No sistema desenvolvido o índice utilizado para validar as comparações pelo método da correlação foi 0,7. Este valor foi obtido através de testes práticos que indicaram ser esta a melhor escolha. Portanto se dois sinais de ECGs, em análise e em memória, forem comparados e o valor da correlação for igual ou maior que 0,7 este sinal terá uma correlação direta, e será atribuída a cardiopatia do ECG em memória para o ECG em análise. Por outro lado se a comparação entre estes sinais for menor que 0,7, não haverá correlação e nenhuma atribuição será feita para o ECG em análise.

Capítulo 6

Funcionamento do Sistema

6.1. Considerações iniciais

Conforme apresentado no Capítulo 2, o sistema proposto tem duas fases de implementação a de hardware, através do XPS, e a de software, através do SDK. A versão da suíte utilizada neste trabalho é a XILINX EDK 10.1 com licença de uso para a Universidade Federal de Itajubá. Portanto toda a construção do sistema proposto segue a metodologia de projeto de sistemas embarcados da XILINX. Neste capítulo, será abordada a parte do SDK, o software construído, e em seguida o XPS, o hardware utilizado.

6.2. Desenvolvimento do *Software*

Todo o sistema foi desenvolvido em linguagem C. O diagrama de blocos, com todas as principais funcionalidades do programa, é detalhado na Figura 6.1.

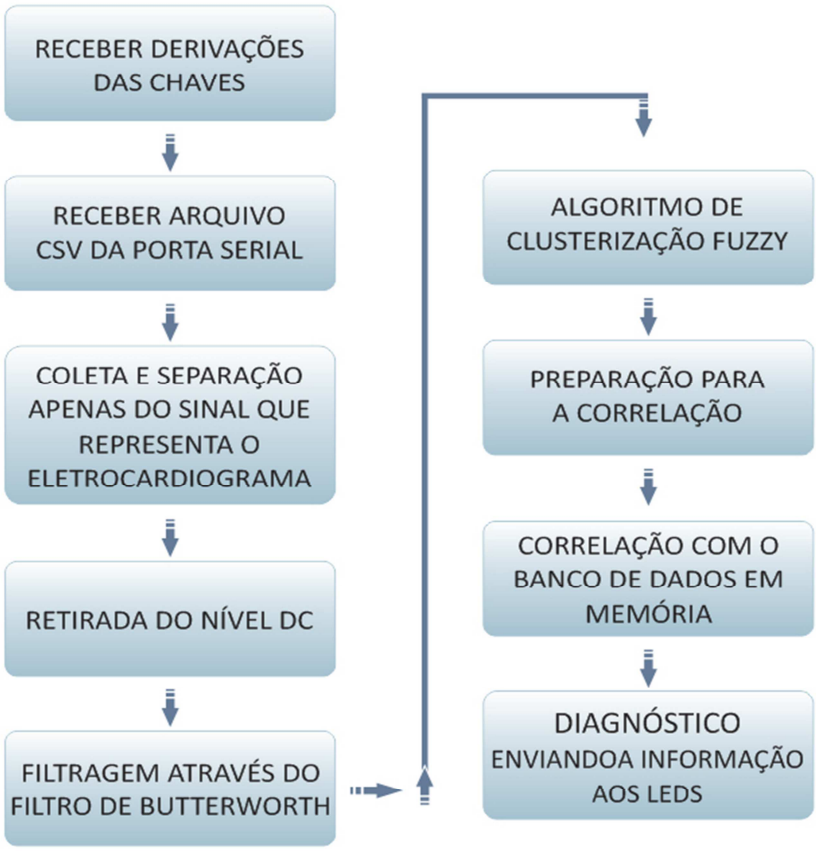

Figura 6.1 – Diagrama de blocos do programa.

As principais dificuldades no desenvolvimento de um software para um sistema embarcado são as limitações de todo o hardware, incluindo os periféricos, e os algoritmos que devem apresentar o melhor desempenho possível.

Não foi inserido nenhum sistema operacional e o hardware também não possui nenhum sistema de gerenciamento de memória, portanto este deve ser feito no próprio código. A alocação, liberação e reuso, principalmente de vetores e matrizes, muito utilizadas nos algoritmos deste sistema, devem ser eficientes, visto que com a falta de gerencia de memória estes dados podem ser corrompidos por outras variáveis.

O sistema é dedicado, ou seja, executa apenas os algoritmos construídos em linguagem C e carregados na memória do Microblaze. A não existência de um sistema operacional, por exemplo, cria dificuldades nas etapas de projeto e implementação, entretanto favorece o desempenho de um sistema embarcado que visa à máxima eficiência.

Ao final da compilação dos arquivos C, o SDK gera um arquivo ELF que é enviado a memória de inicialização do Microblaze, que neste trabalho utilizou-se a memória DDR2 SDRAM. Este arquivo ELF contém os códigos de controle do processador baseados no algoritmo C implementado. Com o envio do ELF, o sistema torna-se operacional.

6.3. Desenvolvimento do *Hardware*

As necessidades de *hardware* foram projetadas e implantadas baseadas no software proposto para a solução do problema de diagnóstico de doenças do coração. Todos os IP cores do XPS, citados neste trabalho, são configurados na FPGA juntamente com o Microblaze. A Tabela 6.1 exemplifica o *hardware* utilizado.

Tabela 6.1 – Hardware Utilizado no Sistema.

Classificação	**Nome**	**Nome na Arquitetura**	**Versão do IP**	**Funções básicas no Sistema**
Processador	Microblaze	Microblaze_0	7.10.d	Rodando no máximo a 50 MHz de frequência. Interpreta as instruções do software e efetua as operações lógicas e aritméticas pertinentes.
Memória Principal	DDR2 SDRAM	DDR2_SDRAM	4.03.a	Armazena instruções e dados durante a execução do programa.
Dispositivo de Entrada	Botões	BTNs_4Bit	1.00.a	Servem apenas para reiniciar o sistema.
Dispositivo de Entrada	Chaves	DIPs_4Bit	1.00.a	Representam a derivação do eletrocardiograma a ser recebido pelo sistema.
Dispositivo de Saída	Leds	LEDs_8Bit	1.00.a	Indicam no 1º estágio a derivação escolhida e no 2º estágio o diagnóstico.
Dispositivo de Saída	Contador de Clock	xps_timer_1	1.00.a	É um *timer* que monitora os pulsos de clock do processador.
Dispositivo de Entrada e	Porta RS-232	RS232_DCE	1.00.a	Recebe o sinal do eletrocardiograma e envia

Saída	(Serial)			informações sobre o sistema.

Para alguns *hardwares* uma explicação mais detalhada será fornecida devido a suas especificidades e relevâncias na funcionalidade do sistema.

6.3.1 Escolhendo as Derivações a Serem Recebidas

Conforme o item anterior, as chaves são as responsáveis pela seleção da derivação de entrada. Há quatro chaves nomeadas na placa, respectivamente, de sw3, sw2, sw1, sw0. Neste sistema considerou-se o sw3 como o bit menos significativo e o sw0 como o bit mais significativo, conforme ilustra a Figura 6.2.

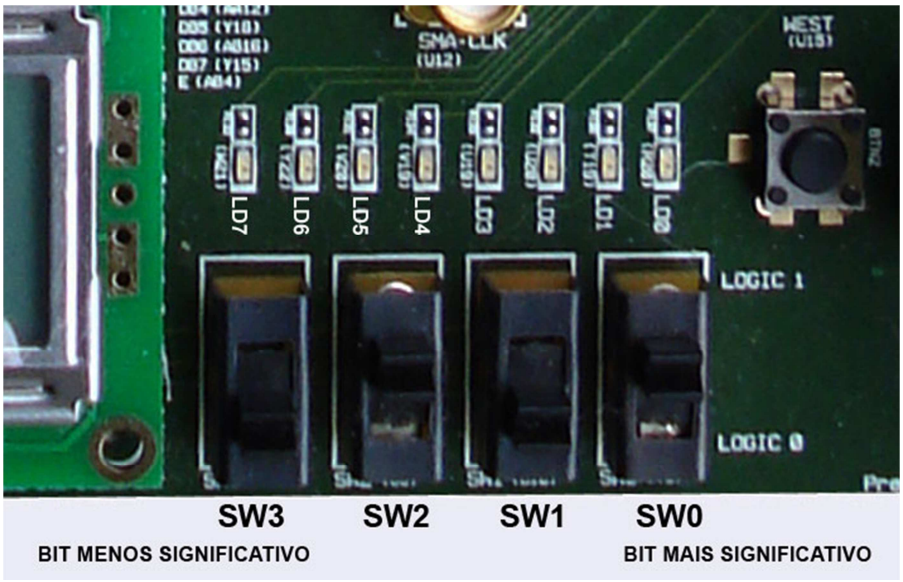

Figura 6.2 – Configuração da entrada das derivações.

O sistema trabalha com sete derivações, que são as disponíveis no banco de dados utilizado. Estas derivações foram classificadas em números inteiros conforme a Tabela 6.2.

Tabela 6.2 – Classificação das Derivações.

Derivação	**Classificação**
MLI	1
MLIII	3
V1	7
V2	8
V3	9
V4	10
V5	11

Para, por exemplo, selecionar a derivação V4, colocam-se as chaves na respectiva posição de classificação que é 10, conforme ilustra a figura 6.3.

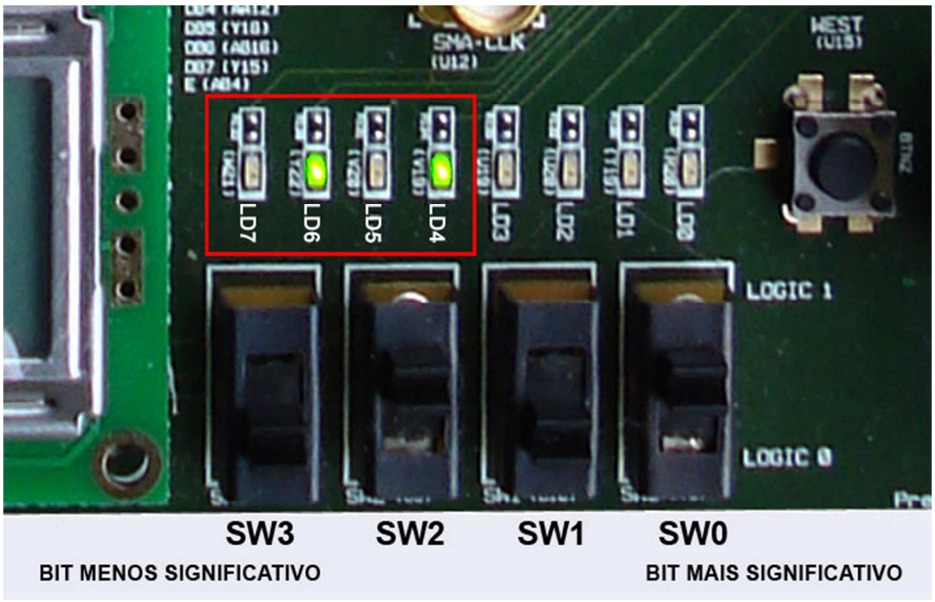

Figura 6.3 – Exemplo das chaves selecionando a derivação V4.

6.3.2 Recebendo o Sinal Amostrado

Foi criado um sistema de comunicação via Hyper Terminal com a interface RS-232 da placa. É através do Hyper Terminal que são enviados os sinais para o sistema. O sistema foi configurado para receber diretamente os arquivos CSV (códigos separados por vírgula) adquiridos automaticamente, em tempo real, do site da PHYSIONET (http://www.physionet.org).

O eletrocardiograma utilizado neste trabalho é o European ST-T Database (EDB) [18], (http://www.physionet.org/cgi-bin/ATM), conforme indicado na Figura 6.4.

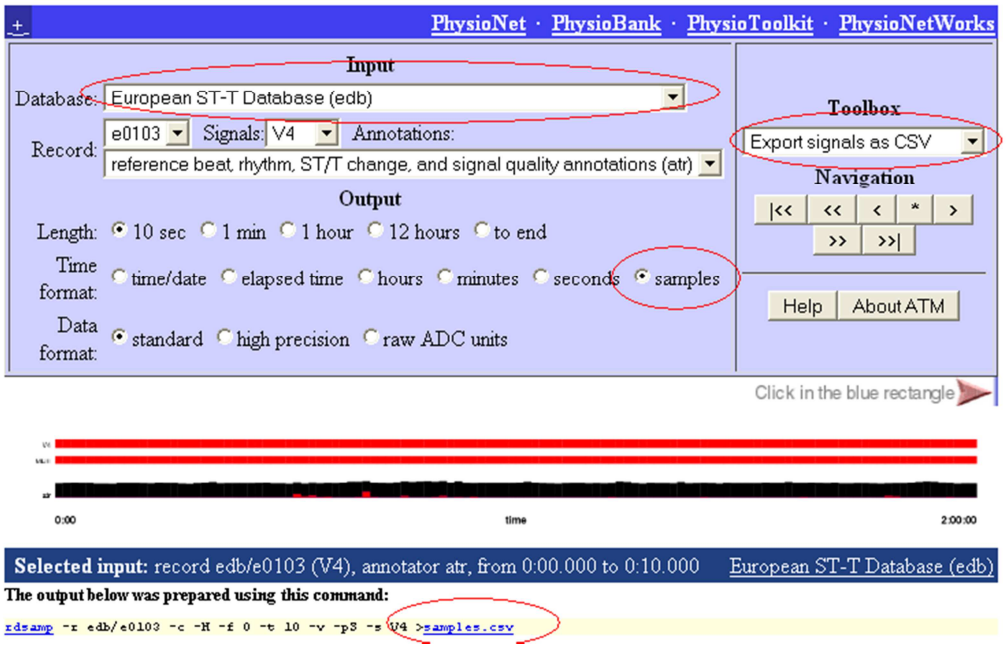

Figura 6.4 – Parâmetros necessários para a aquisição das amostras.

Os círculos em vermelho na Figura 6.4 indicam os seguintes parâmetros:

Database – Base de dados utilizada no sistema, no caso a “European ST-T Database (edb)”;

Time Format – Indicação da 1ª coluna, no sistema marcar a opção “*samples*” (amostras);

ToolBox – Várias opções para o sinal, “*Export signals as CSV*”;

samples.csv – link para o arquivo CSV, amostra pronta para a entrada no sistema.

É importante destacar também os seguintes parâmetros da Figura 6.4

Record – Seleção entre todas as amostras disponíveis para essa base de dados;

Signals – Seleção da Derivação que será gerada.

Concluída a etapa de aquisição do arquivo CSV é necessário enviá-la ao sistema, onde foi utilizado o Hyper Terminal, exemplificado na Figura 6.5.

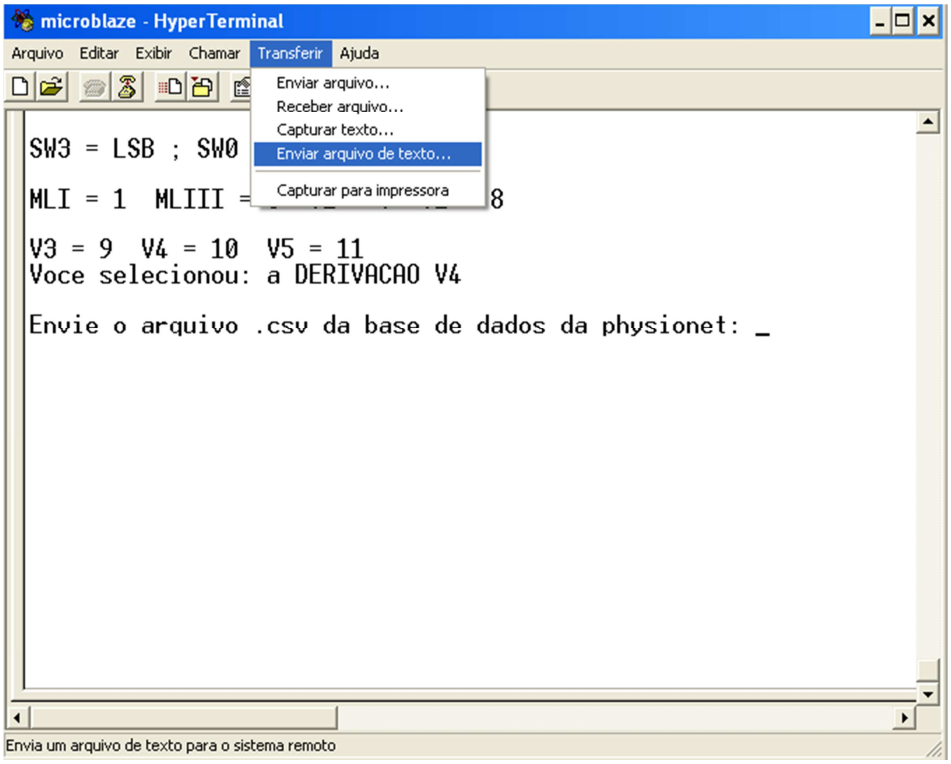

Figura 6.5 – Enviando o sinal para o sistema.

6.3.3 Saídas do Sistema

As saídas do sistema são praticamente baseadas em *LEDs*. A placa possui oito *LEDs* com a seguinte nomenclatura ld7, ld6, ld5, ld4, ld3, ld2, ld1, ld0, conforme a Figura 6.6.

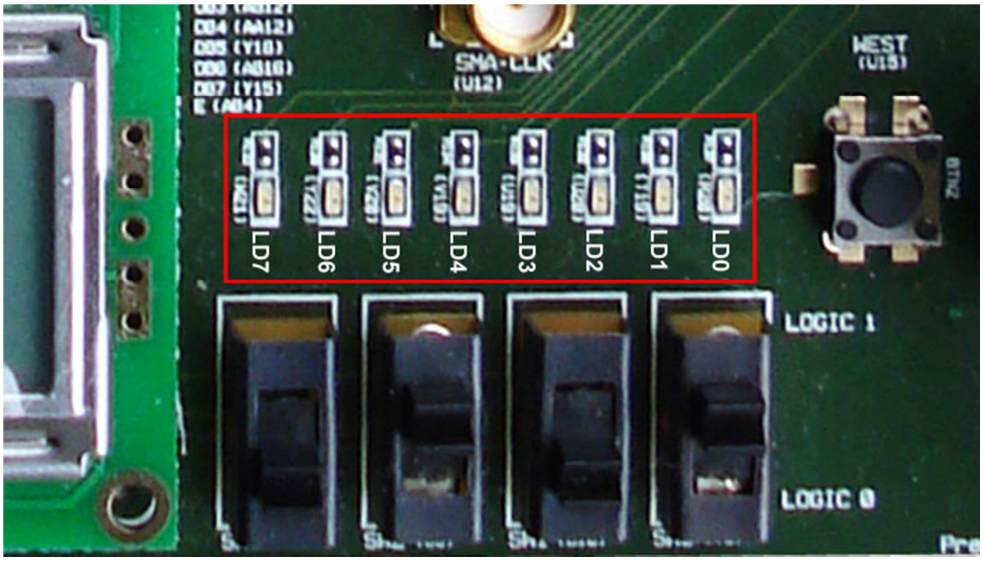

Figura 6.6 – Identificação dos *LEDs*.

Os *LEDs* fornecem dois tipos de informação. A primeira indica a derivação selecionada e a segunda indica o possível diagnóstico para o sinal inserido no sistema. A Tabela 6.3 indica os *LEDs* ativos por derivação e a Tabela 6.4 indica os *LED*s ativos pelo diagnóstico.

Tabela 6.3 – LEDs Ativos pela derivação escolhida.

Classificação	**Derivação**	***Leds* Ativos**
1	MLI	Ld7
3	MLIII	Ld7, ld6
7	V1	Ld7, Ld6, Ld5
8	V2	Ld4
9	V3	Ld7, Ld4
10	V4	Ld6, Ld4
11	V5	Ld7, Ld6, Ld4

Tabela 6.4 – Diagnóstico final do sistema.

Diagnóstico	***LEDs* Ativos**
Infarto	Ld0
Angina	Ld1
Doença arterial coronariana	Ld2
Hipertensão arterial	Ld3
Nenhum	Ld7

Como o exemplo, a placa da Figura 6.7 indica a derivação V4, e na Figura 6.8 têm-se a indicação de um possível diagnóstico, angina.

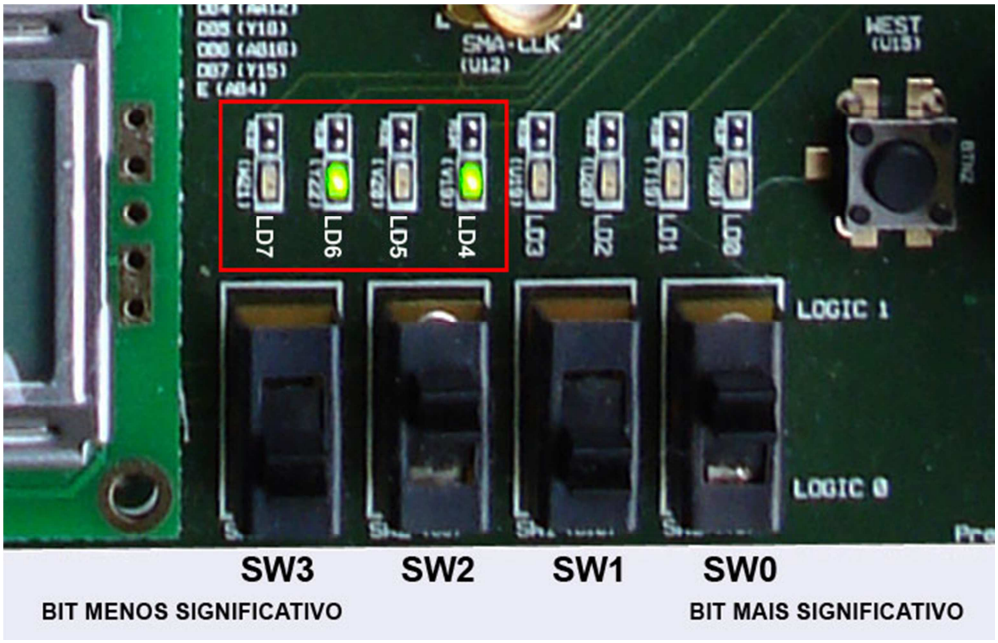

Figura 6.7 – Derivações exibidas nos LEDs (1º momento).

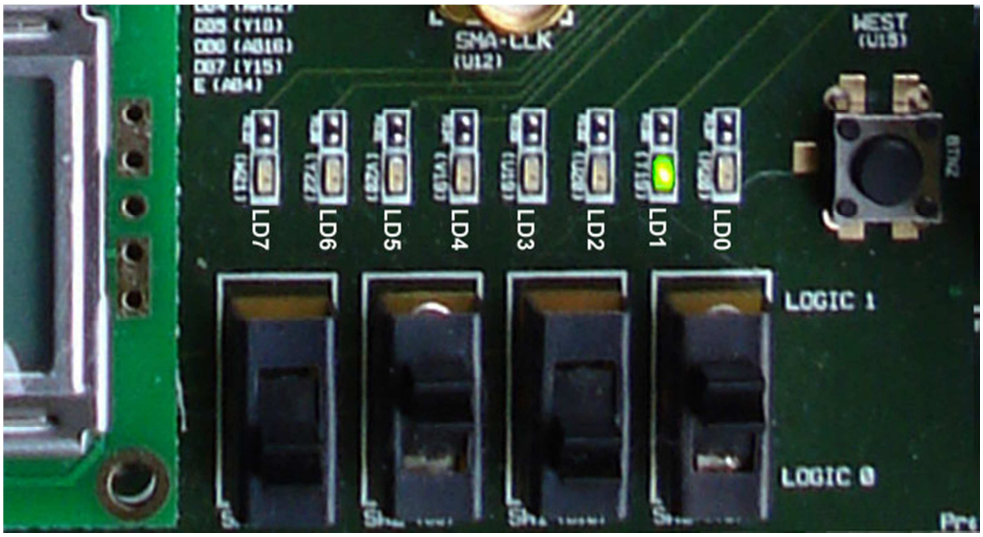

Figura 6.8 – Diagnóstico exibido pelos *LEDs* (2º momento), neste caso angina.

A Figura 6.9 indica as cardiopatias consideradas para diagnóstico.

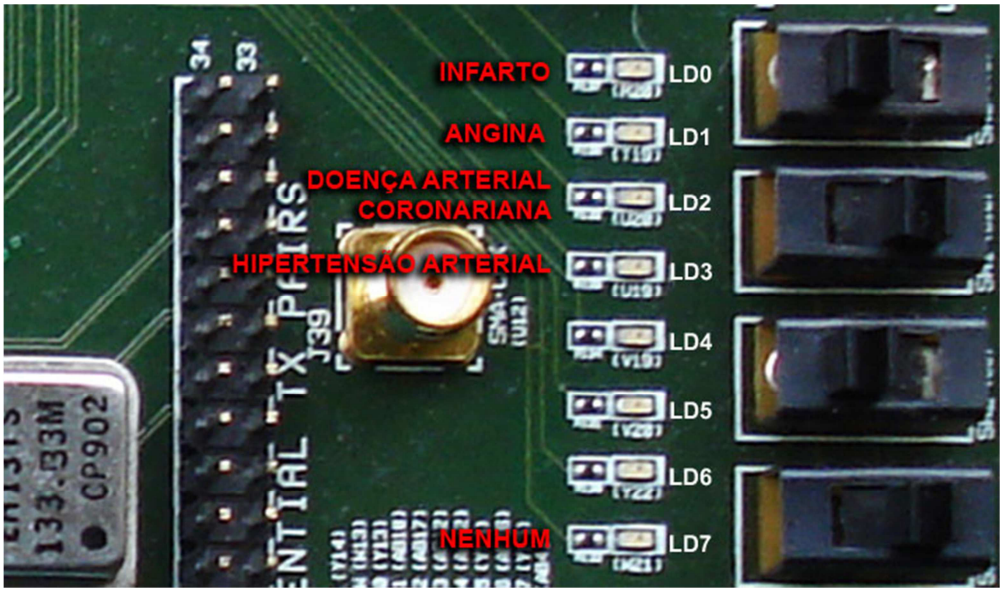

Figura 6.9 – Indicação das cardiopatias pelos *LEDs*.

6.3.4 Hardwares para testes e verificações

Algumas funcionalidades do sistema foram implementadas apenas para validações e verificações como é o caso do contador de *clock* e da interface de saída da porta RS-232.

A função do "contador de clock" é medir quantos ciclos de *clock* de processador são gastos para executar determinada tarefa. Isso foi muito importante na fase de implementação do software, pois indicava se o algoritmo estava, ou não, com um desempenho satisfatório. Validou também o algoritmo de *Fuzzy Clustering* que será apresentado no Capítulo 7.

A saída da porta RS-232 contribuiu para uma análise mais profunda do sistema, visto que vários parâmetros podem ser examinados, conforme indicado na Figura 6.10.

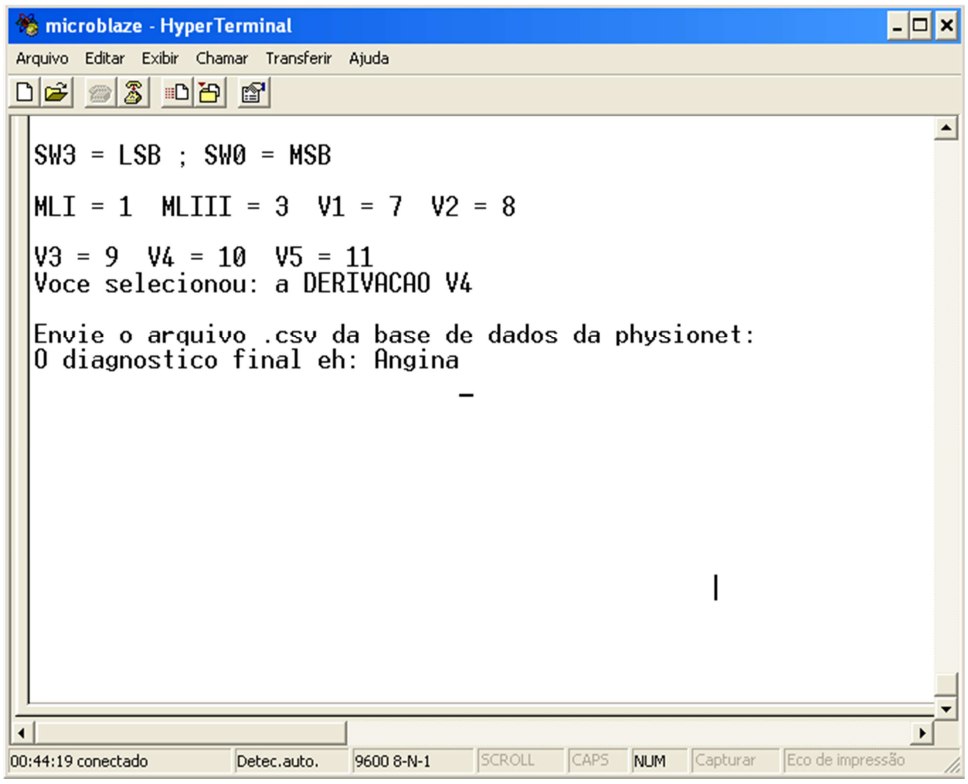

Figura 6.10 – Saída da porta RS-232.

É importante ressaltar que nenhum destes componentes é necessário para o funcionamento do sistema. Em uma futura versão estas funcionalidades podem ser mantidas apenas para modos de checagem, programação e atualização do sistema, como por exemplo, a atualização da base de dados. Para todos os testes realizados e validação do equipamento estes dispositivos foram muito úteis.

Capítulo 7

Resultados Obtidos

7.1. Banco de Dados

De acordo com o processamento proposto, fez-se necessária a criação de um banco de dados de sinais de eletrocardiogramas com diagnóstico conhecido. No banco de dados montado para este trabalho foram armazenados 50 sinais com 213 amostras cada um. Essas 213 amostras representam 1 ciclo completo para analise do eletrocardiograma.

Cada sinal do banco de dados passou pelo pré-processamento e depois foram armazenados apenas os clusters ou os pontos característicos de cada sinal. Esta armazenagem é feita em memória no momento em que o arquivo ELF é carregado no Microblaze.

7.2. Testes Realizados

De acordo com os artigos estudados, todos utilizaram como sinais para validar o sistema, o banco de dados European ST-T, fornecido pelo PhysioNet. Estes foram recebidos através dos arquivos CSV, conforme demonstrado no Capítulo 6.

Antes de realizar os testes, foram verificados outros artigos [12], [13] e [16], que através de outras técnicas e utilizando o mesmo banco de dados, forneceram algumas figuras de mérito, que são os parâmetros Se, *sensitivity* – sensibilidade, e PPV, *Positive Predictive Value* - Valor preditivo positivo ou *precision rate* - taxa de precisão, calculados da seguinte forma [29] e [30]:

$$\text{Se} = \frac{Tp}{Tp + Fn} \tag{7.1}$$

Onde Tp e Fn são os diagnósticos corretos e os diagnósticos que não foram detectados respectivamente.

O parâmetro Se indica a porcentagem de diagnósticos corretos em relação aos diagnósticos não detectados.

$$PPV = \frac{Tp}{Tp + Fp} \tag{7.2}$$

Onde Fp indica os diagnósticos errados fornecidos pelo método.

O parâmetro PPV indica a porcentagem de diagnósticos corretos em relação aos diagnósticos errados fornecidos pelo sistema em teste.

7.3. Resultados Obtidos

De acordo com a ferramenta matemática apresentada e os parâmetros a serem comparados, foram feitos testes para verificação da eficácia da ferramenta. Para isto, foi montado um banco de dados formado por amostras de sinais de eletrocardiogramas com características de algumas cardiopatias como angina, infarto e hipertensão arterial.

Foram feitos 37 diagnósticos de pacientes diferentes, e com diagnósticos também diferentes. Como já apresentado cada sinal, antes de ser comparado com o banco de dados, passou por um processo de filtragem, onde neste sistema foi utilizado um filtro Butterworth digital de terceira ordem passa baixa, para eliminar ruídos de alta frequência e nivelamento do sinal, para eliminar ruídos de níveis DC que possam estar juntos com o sinal do eletrocardiograma.

De acordo com os artigos estudados e o resultado dos testes realizados neste trabalho, elaborou-se a Tabela 7.1 que compara os parâmetros Se e PPV.

Tabela 7.1 – Comparação de resultados

Sistema	Se (%)	PPV(%)
Taddei	84	81
Vila	83	75
Jager	87	88
Maglaveras	89	78
Andreão	83	86
Fuzzy Cluster Algorithm	**75**	**92**

Na Tabela 7.1 observa-se que no sistema proposto o parâmetro Se, de 75%, ficou abaixo, se comparado com os outros sistemas, o que significa que o sistema apresentado não tem um bom desempenho quando analisada a possibilidade de não detectar uma determinada cardiopatia. Entretanto o parâmetro *Se* ainda ficou dentro de limites aceitáveis. Em relação ao parâmetro *PPV,* de 92%, o sistema proposto teve um desempenho melhor do que os outros sistemas, o que confirma a eficácia quando o sistema detecta uma cardiopatia pois neste caso a chance de indicar uma cardiopatia errada é bem menor.

Além de apresentar um bom resultado se comparado com outros sistemas, esta técnica de *Fuzzy Clustering* permitiu, com apenas 20 pontos de cada sinal, fazer o diagnóstico de cardiopatias. Consequentemente o sistema ocupa um menor espaço em memória para armazenar o banco de dados e tem um processamento menor, gerando o diagnóstico de forma mais rápida. Isto pode ser comprovado com o “Contador de clock”, que forneceu os dados exibidos no gráfico da Figura 7.1.

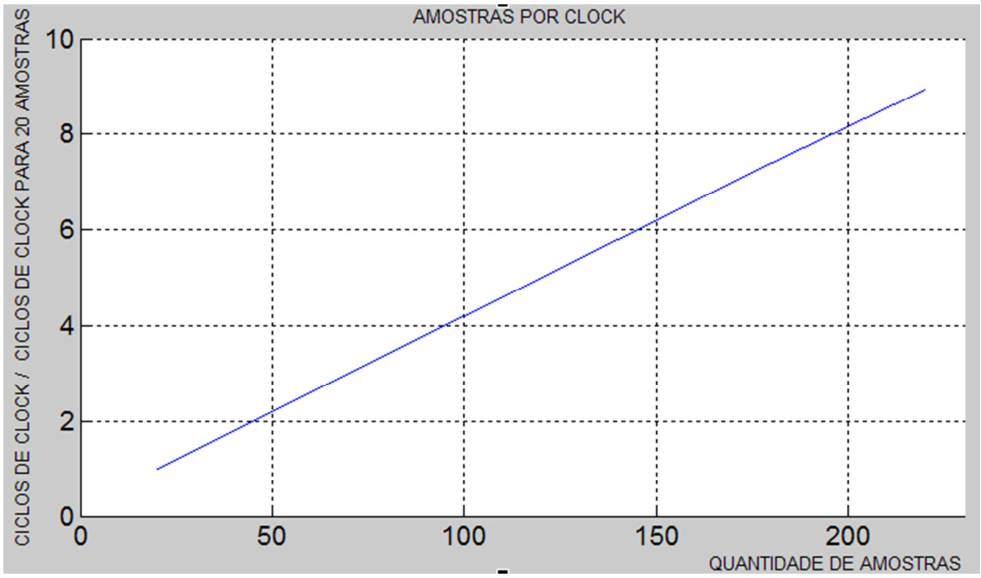

Figura 7.1 – Gráfico da quantidade de *clocks* por amostra.

Foi feita uma normalização nos ciclos de *clock* para 20 amostras, sendo representado por 1 indicado no gráfico da Figura 7.1. Neste gráfico observa-se que para **cada ciclo de *clock*** gasto na execução da operação de correlação para o sinal do ECG completo, 213 amostras, são gastos aproximadamente 9 vezes menos ciclos de *clock* para o sinal do ECG com a aplicação do *Fuzzy Clustering*, ou seja, 20 amostras. Este dado é interessante visto que o ganho computacional é elevado, isto no caso de um algoritmo de comparação como a correlação, onde devido a suas características o *clock* cresce ou decresce linearmente. Como a correlação é uma técnica de comparação a quantidade de amostras por ciclo de *clock* foi, como esperado, linear. Entretanto em outras técnicas de detecção de ECG, tais como redes neurais em [17], a melhora de desempenho tende a ser exponencial, pois a quantidade relativa de dados de entrada é muito menor e pode reduzir o tratamento e melhorar a resposta do sistema.

Este sistema funciona na frequência máxima de 50 MHz. Esta frequência quando comparada às de outros trabalhos como, por exemplo [31] que trabalha em 200 MHz, se demonstra eficaz para um sistema de diagnóstico de ECG embarcado.

Capítulo 8

Conclusões e Trabalhos Futuros

Um sistema embarcado com a solução desenvolvida por este trabalho poderá auxiliar médicos e profissionais da área da saúde na tomada de decisões. Este sistema não tem pretensões, sob nenhuma hipótese, de substituir estes profissionais, porém para médicos recém-formados, ou talvez em situações de urgência e emergência, o mesmo poderá ser de grande valia. Também é possível fornecer prováveis diagnósticos para, no início, auxiliar os cardiologistas, principalmente em regiões fora dos grandes centros onde os recursos são poucos e não possuem, às vezes, especialistas da área ou em plantões médicos onde é muito comum o médico de plantão não ser um especialista e, nesse caso, o equipamento poderia auxiliá-lo ao fazer um diagnóstico de cardiopatias.

Com relação aos métodos empregados na implementação deste sistema pôde-se verificar uma melhora em relação aos outros sistemas apresentados no que diz respeito a possibilidade de fornecimento de diagnóstico correto, com isso têm-se uma maior confiança nos resultados apresentados, principalmente quando este resultado acusa uma possível cardiopatia. Esta conclusão está relacionada com o parâmetro *PPV*. Se for considerado o parâmetro *Se*, conclui-se que o resultado, apesar de apresentar um baixo desempenho em relação à omissão de um diagnóstico, este item não terá tanta influência para um sistema onde

pretende-se ter um resultado com um mínimo de possibilidade de erro de diagnóstico, além deste ponto de vista, o sistema pode ter o parâmetro Se melhorado se o banco de dados possuir mais sinais, com características diferentes.

Outro resultado importante a se destacar, em relação às técnicas empregadas, é que se comparado com os resultados de algoritmos de testes feitos em outras referências, os resultados obtidos utilizando a *Fuzzy Clustering* e a correlação foram tão bons ou melhores que algumas figuras de mérito. Este fato justifica o interesse em aperfeiçoar e utilizar estes métodos.

Para que o sistema tenha aplicações reais têm-se várias opções como, por exemplo, a criação de circuitos analógicos para as etapas de pré-processamento e filtragem. É necessário também um conversor analógico digital para inserir os sinais amostrados e filtrados no sistema. Desta forma testes em tempo real, com equipamentos e pacientes reais, e principalmente o acompanhamento de um especialista em cardiologia garantiriam o aperfeiçoamento e o futuro deste projeto. A inserção de dados sintomáticos relatados pelo paciente, em uma consulta clínica, pode ajudar a distinguir entre uma ou outra cardiopatia, melhorando ainda mais a eficiência e confiabilidade do sistema desenvolvido.

Com este *hardware*, e as técnicas utilizadas, abrem-se várias linhas de pesquisa, para a análise e identificação de outros sinais biológicos tais como sinais cerebrais e musculares. A implementação em *hardware*, e o uso das técnicas empregadas neste trabalho, melhoram o desempenho da arquitetura no sentido em que diminuem o tempo de processamento e os requisitos com relação à capacidade de armazenamento.

Outra grande vantagem desta implantação em *hardware* é o custo. A placa utilizada para o desenvolvimento do sistema é uma XILINX SPARTAN 3A Starter Kit. Esta é cotada, no sítio da XILINX [32], em 189,00 dólares americanos, o que comprova o baixo custo do sistema desenvolvido, visto que mesmo esta não tem todos os seus recursos necessários e explorados. Este valor serve apenas como referência visto que na produção em larga escala, e somente da FPGA, este circuito pode ter valores ainda mais reduzidos.

Tem-se ainda a possibilidade de validar este sistema diretamente na FPGA usando as linguagens de descrição de h*ardware*, HDLs. Desta forma poderia estar sendo viabilizado um *Hard CPU Core*, dispensando o uso do IP, Microblaze, e de todo o *software* para este sistema, diminuindo ainda mais os custos, aumentando a eficiência do sistema e criando-se realmente um processador dedicado de sinais de eletrocardiograma.

Referências Bibliográficas

[1] National Center for Health Statistics (2009). Health, United States: With Special Feature on Medical Technology. Hyattsville. Retrieved March 05, 2009, from http://www.cdc.gov/nchs/data/hus/hus09.pdf

[2] Carter M. (2006). Heart disease still the most likely reason you'll die. CNN, A Time Warner Company. Retrieved January 05, 2008, from http://edition.cnn.com/2006/HEALTH/10/30/heart.overview/index.html

[3] Sociedade Brasileira de Cardiologia (2003). Orientações para a interpretação do eletrocardiograma de repouso. Arquivos Brasileiros de Cardiologia, Acessado em 07 de janeiro de 2008 em: http://publicacoes.cardiol.br/consenso/2003/8002/repouso.pdf

[4] Negreiros de Andrade, PJ. (3rd Ed.). (2008). Cardiologia para o Generalista: Uma abordagem fisiopatológica. Brasil/Fortaleza, Ceará: UFC.

[5] Abreu-Lima C, de Sa JP. Automatic classifiers for the interpretation of electrocardiograms. Rev Port Cardiol 1998; 17(5):415-28.

[6] Willems JL. Quantitative electrocardiography. Standardization and performance evaluation. Ann N Y Acad Sci 1990; 601(1):329-42.

[7] RuDusky BM. Errors of computer electrocardiography. J Vascular Diseases 1997; 48(12):1045-50.

[8] Nianqiang, L., & Yongbing, W., & Guoyi, Z. (2010). A Preferable Method on Digital Filter in ECG Signal's Processing Based on FPGA. Third International Symposium on Intelligent Information Technology and Security Informatics, 184-187.

[9] Mitsukura, Y., & Miyata, K., & Mitsukura, K., & Fukumi M., & Akamatsu, N. (2004). Intelligent Medical Diagnosis System Using the Fuzzy and Neural Networks. IEEE Annual Meeting of the North American Fuzzy Information Processing Society, Vol 2, 550-554.

[10] Yan, L., & Hang, Y., & Lai, J., & Lixiao, M., & Zhen, J. (2010). Adaptive Lifting Scheme for ECG QRS Complexes Detection and its FPGA implementation. Third International Conference on Biomedical Engineering and Informatics, 721-724.

[11] Jewajinda, Y., & Chongstitvatana, P. (2010). FPGA-based online-learning using parallel genetic algorithm and neural network for ECG signal classification. Seventh International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), 050-1054.

[12] Andreão, R., V. (2004). T-segment analysis using hidden Markov Model beat segmentation: application to ischemia detection. Computers in Cardiology, 381-384.

[13] Jager, F., & Moody, G., & Mark, R. (1998). Detection of transient ST segment episodes during ambulatory ECG monitoring. Computers and biomedical research an international journal, Vol 31, 305-322.

[14] Maglaveras, N., & Stamkopoulos, T., & Pappas, C., & Gerassimos Strintzis, M. (1998). An adaptive backpropagation neural network for real-time ischemia episodes detection: development and performance analysis using the European ST-T database. IEEE Transactions on Biomedical Engineering, Vol 45, 805-813.

[15] Taddei, A., & Constantino G. (1995). A System for the Detection of Ischemic Episodes in Ambulatory ECG. Computers in Cardiology, 705–708.

[16] Vila, J., & Presedo, J., & Delgado, M., & Barro, S., & Ruiz R., & Palacios, F.

(1997). SUTIL: Intelligent ischemia monitoring system. International Journal of Medical Informatics 47, 193–214.

[17] Armato, A., & Nardini, E., & Lanatà, A., & Valenza, G., & Mancuso, C., & Scilingo, E., & Rossi, D. (2009). An FPGA based arrhythmia recognition system for wearable applications. Ninth International Conference on Intelligent Systems Design and Applications, 660-664.

[18] Goldberger, AL., & Amaral, L., & Glass, L., & Hausdorff, JM., & Ivanov, PCh., & Mark, RG., & Mietus, JE., & Moody, GB., & Peng, C-K., & Stanley, HE. (2000). PhysioBank, PhysioToolkit, and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals. Retrieved March 10, 2009, from http://physionet.org/physiobank/database/edb/

[19] Fuganti, Cláudio José; Oliveira, Divina Seila; Rodrigues, Ricardo José; “Curso de Eletrocardiografia Básica”; Ccs – Centro de Ciências da Saúde; UEL – Universidade Estadual de Londrina.

[20] [20] Stein, E. (1st Ed.) (1987). Clinical Electrocardiography: A Self-Study Course. USA/Philadelphia, PA: Lea & Febiger.

[21] Oliveira, A., & Andrade, F. (1st Ed.). (2006). Sistemas Embarcados: Hardware E Firmware Na Prática. São Paulo, SP: Editora Érica.

[22] Zeidman, B. (1st Ed.). (2002). Designing with FPGAs and CPLDs. London: CMP Publishing.

[23] Xilinx Inc (2009). MicroBlaze Soft Processor Core. Acessado em 10 de março de 2009, from http://www.xilinx.com/tools/microblaze.htm

[24] Boylestad, R., & Nashelsky, L., & Monssen, F. (9th Ed.). (2005). Electronic Devices and Circuit Theory. USA/ Upper Saddle River, NJ: Prentice Hall.

[25] Ian, S., & Simões, M. (2nd Ed.). (2007). Controle e Modelagem Fuzzy. Brazil/São

Paulo, SP: Editora Edgard Blucher Ltda.

[26] M. Setnes, “Supervised Fuzzy for Rule Extraction”, IEE TRANSACTIONS ON FUZZY SYSTEMS, Vol. 8, N° 4, August 2000.

[27] Gustafson, D., & Kessel, W. (1979). Fuzzy clustering with a fuzzy covariance matrix. Hemometrics and Intelligent Laboratory Systems - IEEE, 761-766.

[28] Bendat, J., & Piersol, A. (2nd Ed.). (1993). Engineering Applications of Correlation and Spectral Analysis. NY, NY: John Wiley & Sons.

[29] Altman, D., & Bland, M. (1st Ed.). (1994). Statistics Notes: Diagnostic tests 1: sensitivity and specificity. London: BMJ 308, P 1552.

[30] Altman, D., & Bland, M. (1st Ed.). (1994). Statistics Notes: Diagnostic Tests 2: Predictive Values. London: BMJ 309, P 102.

[31] Yan, L., & Hang, Y., & Lai, J., & Lixiao, M., & Zhen, J. (2010). Adaptive Lifting Scheme for ECG QRS Complexes Detection and its FPGA implementation. Third International Conference on Biomedical Engineering and Informatics, 721-724.

[32] Xilinx Inc (2010). Spartan-3A Starter Kit. Acessado em 30 de setembro de 2010, de http://www.xilinx.com/tools/microblaze.htm