# Simplificações para Redução do Custo Computacional da Pré-ênfase de Voz na Plataforma Arduino

**Emerson B. da Cunha[1], Pedro Ítalo R. Albuquerque[1], Daniella Dias C. da Silva[1]**

[1]Instituto Federal de Educação, Ciência e Tecnologia da Paraíba (IFPB) – *Campus* Campina Grande, PB – Brasil

emersonbarbosa.ifpb@gmail.com, pedroitaloifpb@gmail.com, daniella.silva@ifpb.edu.br

***Abstract.*** *Currently, there is a growing interest in applications where human-machine interaction is conducted via the human voice. However, some devices, such as mobile phones and home appliances, have limited capacity of storage and processing, being difficult to implement this type of system in this context. In this study, simplifications were used in one of the stages of the speech recognition process, which were compared and analyzed with respect to the traditional implementation. Therewith, was obtained a reduction of up to one fifth of the runtime and number of clock cycles*.

***Resumo***. *Atualmente, existe um crescente interesse por aplicações em que a interação homem-máquina seja realizada via a voz humana. No entanto, alguns equipamentos, como telefones celulares e eletrodomésticos, possuem limitações de armazenamento e processamento, dificultando a implementação deste tipo de sistema. Neste trabalho, foram utilizadas simplificações em uma das etapas do processo de reconhecimento da fala, as quais foram comparadas e analisadas em relação à implementação tradicional. Diante das mesmas, foi obtida uma redução de até 1/5 do tempo de execução e número de ciclos de clock.*

## 1. Introdução

O ser humano sempre buscou desenvolver formas ou métodos de comunicação variadas, obviamente no nível de conhecimento de cada época. E uma destas formas é a fala, que tem um importante papel na difusão e manipulação de informação, seja desde a simples decodificação de palavras de uma criança a mecanismos tecnológicos cada vez mais sofisticados.

Diante do crescente interesse por aplicações de *softwares* e equipamentos que "compreendam", reconheçam e simulem a voz humana, pesquisas têm sido realizadas na área de Processamento Digital de Sinais de Voz (PDSV) [KLEIJN 1998], [DE LIMA 2000], [BENZEGHIBA 2003], [DA CUNHA 2003], [SILVA 2011] [CIPRIANO 2001]. Atualmente, equipamentos eletrônicos que permitem realizar a iteração homem-máquina por meio do reconhecimento de fala são cada vez mais frequentes, seja na área de segurança (como forma de controle de acesso), mobilidade (indivíduos com limitações físicas), praticidade (acesso a funções de dispositivos como *smartphones*, *tablets*, mídia *center* de veículos automotores, etc.), enfim, existe uma grande diversidade de aplicações na utilização deste tipo de tecnologia. No entanto, esses

equipamentos muitas vezes possuem um *hardware* simples e, consequentemente, limitações na sua capacidade de processamento e armazenamento. Diante disso, torna-se necessário o desenvolvimento de *software* adaptado para este ambiente. Surge então, o conceito de sistema embutido (ou embarcado), que é a combinação de *hardware* e *software* e, algumas vezes, peças mecânicas, desenvolvidos para realizar uma função específica. Por meio da implementação em *hardware*, é possível alcançar maior eficiência e rapidez na execução de determinadas tarefas e, a partir do *software*, reduzir o tempo de desenvolvimento do sistema.

O grande desafio é aliar o bom desempenho ao baixo consumo de energia, possibilitando agregar serviços como o de reconhecimento de fala. Para permitir a implementação de sistemas de PDSV nesse contexto, alguns trabalhos sacrificam a eficiência no processo de reconhecimento em nome da redução da área e exigências computacionais. No entanto, algumas técnicas de programação permitem a geração de códigos mais eficientes no que se refere à quantidade de operações a serem executadas pelo processador como, por exemplo, substituição de uma multiplicação por deslocamento de bits, o que consequentemente também proporciona menor consumo de energia. Sendo assim, a partir da otimização de algoritmos, utilização de estruturas de *hardware* específicas e um paralelismo de instruções eficiente, é possível aumentar a velocidade e reduzir os recursos necessários, sem diminuir a taxa de reconhecimento.

Uma plataforma que tem sido bastante utilizada para o desenvolvimento de sistemas embarcados de maneira rápida é o Arduino. O objetivo deste trabalho foi analisar o impacto no desempenho para esta plataforma de *hardware*, ao se utilizar simplificações matemáticas na etapa de pré-processamento de um sistema de reconhecimento de fala.

Este documento está organizado da seguinte forma. Na Seção 2 são apresentadas as principais características da plataforma Arduino. Na Seção 3 são descritas as etapas típicas de um sistema de reconhecimento de fala. Na Seção 4 são apresentados mais detalhes sobre a etapa de pré-processamento e pré-ênfase, objeto de estudo deste artigo. Na Seção 5 são descritas as simplificações adotadas e as implementações realizadas para testes. Na Seção 6 é realizada uma análise dos resultados obtidos. Por fim, na Seção 7 são apresentadas as conclusões e considerações finais deste trabalho.

## 2. Arduino

A plataforma de *hardware* utilizada neste trabalho foi o Arduino. O mesmo consiste em uma placa fabricada como plataforma de prototipagem eletrônica e que torna a robótica mais acessível a todos. O projeto italiano iniciado em 2005 tinha primeiramente cunho educacional e interagia com aplicações escolares, mas atingiu proporções muito maiores ao longo dos anos, justamente por sua facilidade de uso, onde não é necessário ter tanto conhecimento do *hardware,* como em outros microcontroladores como o PIC, por exemplo.

Também é possível aumentar a capacidade do *hardware* do Arduino, acoplando ao mesmo os chamados *Shields* (escudos) que consistem de placas que são acopladas à placa original, agregando funcionalidades à mesma. Há diversos tipos de *shields*, com as mais variadas funções, por exemplo: comunicação com rede Ethernet, transmissores via-Bluetooth, módulos Zigbee, entre outros.

No entanto, esta vantagem de se trabalhar em alto nível, pode se tornar uma grande desvantagem, quando se é necessário trabalhar no nível de *hardware*, e ter um controle maior sobre como as operações são executadas e controladas. Por exemplo, quando se precisam utilizar soluções de otimizações, como uso de registradores específicos e *pipeline*.

O modelo utilizado neste trabalho foi o Arduino UNO (Figura 1) que é equipado com o microcontrolador ATmega328 da ATMEL®. Ele tem um *clock* base de 16 MHz e vários pinos de entrada e saída, tanto digital quanto analógica, além de módulo PWM (*Pulse Width Modulation),* que já vem implementado nativamente e com funções específicas na própria IDE, o que facilita muito a implementação de aplicações para processamento de áudio.

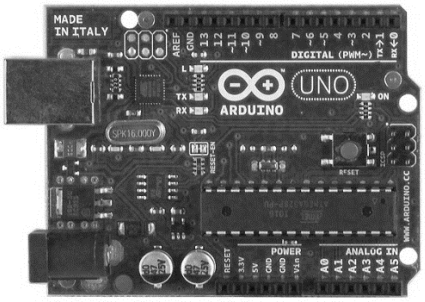


**Figura 1. Arduino UNO.**

## 3. Sistemas de Reconhecimento de Fala

O objeto de estudo deste trabalho é o reconhecimento de fala, então, para que seja possível compreender melhor o objetivo a ser atingido ao final do mesmo, serão apresentadas as etapas típicas deste tipo de sistema (Figura 2).

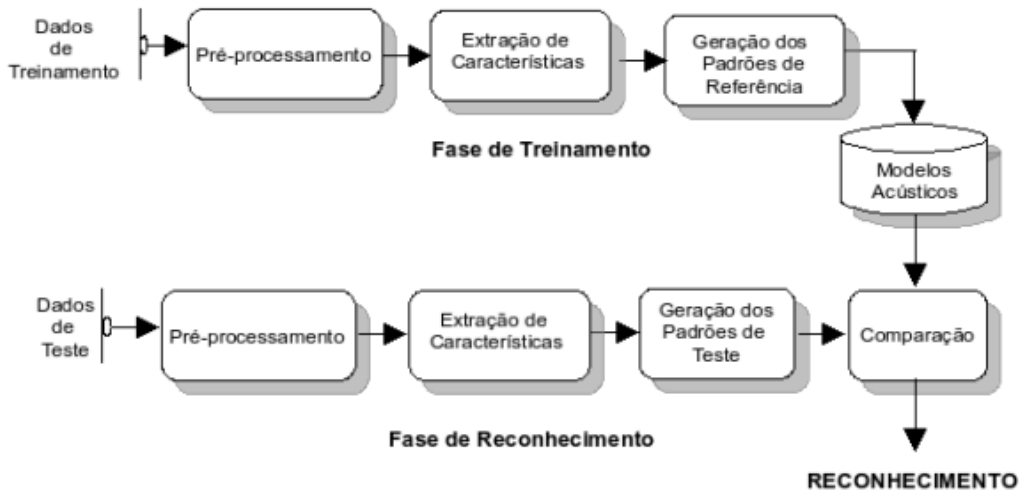


**Figura 2. Sistema de reconhecimento de padrões da fala [Silva 2006].**

Esse tipo de sistema realiza uma tarefa de reconhecimento de padrões, nesse caso padrões de fala, e sempre inclui duas fases: treinamento e reconhecimento. Com base nos dados de treinamento (palavras ou frases), são gerados os modelos de referência (modelo acústico), aos quais são atribuídos rótulos que identificam cada padrão (palavra ou frases). Na fase de reconhecimento, a partir dos dados de teste (sinais de voz) são obtidos padrões de teste que, em seguida, são comparados com os modelos gerados durante o treinamento e, utilizando-se uma regra de decisão, é identificado o que mais se assemelha ao padrão de entrada desconhecido

A etapa de pré-processamento é responsável pelo tratamento do sinal de voz com relação ao ambiente de gravação e canal de comunicação utilizado. O objetivo desse tratamento é reduzir efeitos indesejados incorporados ou presentes no sinal de voz, além de prepará-lo para as etapas seguintes do processo de reconhecimento. Dentre vários aspectos que podem ser tratados nessa etapa, podem ser citados [Rabiner 1978] [Furui 1981] [Shaughnessy 2000]:

- Variações relacionadas ao estilo do falante;
- Ruído no ambiente, no meio de comunicação, etc.;
- Variações no momento de gravação do sinal como, por exemplo, distância entre o locutor e o microfone;
- Considerações quanto ao tipo de unidade da fala a ser processada nas etapas posteriores.

A etapa de extração de características é de extrema importância em sistemas de reconhecimento, uma vez que nesta etapa são obtidos elementos que possibilitam a geração de padrões. Esses elementos também podem ser parâmetros obtidos a partir de modelos de produção de voz. Assim, essa etapa pode ser chamada de extração de parâmetros [Silva 2006].

Durante a fase de treinamento, com base nas características extraídas, são gerados padrões de referência, os quais serão comparados com o padrão de teste na fase de reconhecimento.

Vale salientar que as etapas de pré-processamento e extração de características das fases de treinamento e reconhecimento devem ser equivalentes, para que seja possível a correta comparação entre o padrão testado e o(s) padrão(ões) já conhecido(s) pelo sistema [Silva 2006] [CIPRIANO 2001].

## 4. Pré-processamento Digital de Sinais de Voz

Os sinais de voz são compostos de uma sequência de sons que servem como uma representação simbólica da mensagem produzida pelo locutor para o ouvinte. O sinal de voz produzido pelo homem é naturalmente analógico e para permitir seu processamento por computadores digitais é necessário que seja realizada uma conversão analógico-digital (A/D) sobre o mesmo. O passo seguinte à amostragem é a quantização do sinal amostrado, que se refere à discretização da intensidade (amplitude) do sinal, permitindo a sua representação por uma quantidade finita e pré-definida de bits, obtendo-se então um sinal digital.

Quanto maior a quantidade de bits utilizada nesse processo, maior o número de valores permitidos e menor o de aproximações necessárias, sendo assim o sinal quantizado mais fiel ao original. Após a aquisição e digitalização do sinal de voz, é realizado o pré-processamento nas amostras, a fim de prepará-las para a extração de seus parâmetros e/ou características. Estes são utilizados no algoritmo de reconhecimento de padrões. O pré-processamento implementado neste trabalho, contemplou a função de pré-ênfase, a qual é descrita na seção a seguir.

### 4.1. Pré-ênfase

A distorção provocada pelos lábios produz uma queda na envoltória espectral de, aproximadamente 6dB/oitava uma vez que o sinal de voz apresenta baixas amplitudes nas altas frequências, essa tendência as torna especialmente vulneráveis ao ruído, comprometendo o processo de reconhecimento [SILVA 2006] [CIPRIANO 2001].

Para solucionar esse problema é aplicado um filtro, de resposta aproximadamente +6dB/oitava, que ocasiona um nivelamento no espectro [PETRY 2000]. A esse processo de tratamento do sinal de voz, dá-se o nome de pré-ênfase.

A função de transferência da pré-ênfase consiste de um sistema de primeira ordem fixo, cuja função é dada por:

$$H(z) = 1 - a.z^{-1},\ 0 \leq a \leq 1 \quad (1)$$

Neste caso, a saída da pré-ênfase Sp(n) está relacionada à entrada S(n) pela Equação (2), dada por [Petry 2000]:

$$Sp(n) = S(n) - \alpha.S(n-1) \quad (2)$$

sendo:

- Sp(n) – amostra pré-enfatizada;
- S(n) – amostra original;
- α – fator de pré-ênfase, 0, 9 ≤α≤ 1.

Um valor típico usado é ∝= 0.95, o que significa 20dB de amplificação para as mais altas frequências.

## 5. Simplificações e Implementação

Para analisar o impacto e, consequentemente, a viabilidade da implementação de técnicas de PDSV na plataforma Arduino, foram utilizadas as simplificações matemáticas referentes à etapa de pré-processamento propostas por [Silva 2006] [CIPRIANO 2001]. Inicialmente, foi analisada a função de pré-ênfase, e em etapas seguintes a este artigo, serão avaliadas as etapas de divisão em quadros e janelamento.

Para a função de pré-ênfase, na Equação (2), o fator α é substituído pelo valor 15/16, correspondente a 0,9375 [Silva 2006] [CIPRIANO 2001]:

$$Sp(n) = S(n) - \frac{15}{16}S(n-1) = S(n) - S(n-1) + \frac{S(n-1)}{16} \quad (3)$$

A modificação apresentada na Equação (3) implica na substituição de uma multiplicação por um número fracionário, por uma divisão por um número do tipo $2^m$, sendo m inteiro, e que pode ser realizada através de um simples deslocamento de *m* bits, nesse caso $m = 4$. Durante a divisão por 16 da Equação (3), é realizado um deslocamento de quatro bits para à direita, repetindo quatro vezes o bit mais significativo, à esquerda. Essa repetição é necessária para que seja mantido o sinal do valor correspondente da amostra de voz.

Na Figura 3 é apresentado o diagrama de blocos do sistema implementado.

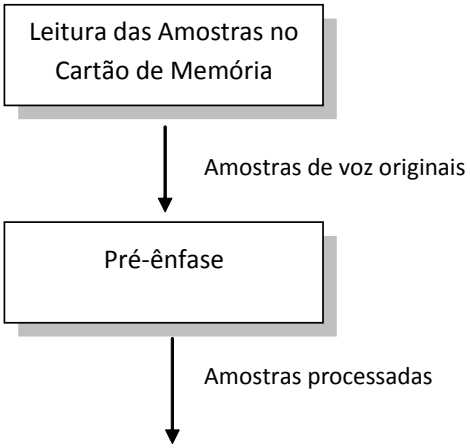


**Figura 3. Diagrama de blocos do sistema implementado.**

Além das versões tradicional e simplificada, a pré-ênfase também foi implementada com e sem módulos. Os módulos em questão consistem na separação da etapa de leitura das amostras de voz, armazenadas em um cartão SD, da função de pré-ênfase propriamente dita.

A versão sem módulo está totalmente implementada em um único arquivo fonte. Por outro lado, a versão com módulo possui uma biblioteca a parte para leitura do cartão SD e outra para a pré-ênfase, sendo estas duas utilizadas em um programa principal. A vantagem da modularização é que, no futuro, caso se deseje captar o sinal de voz diretamente de um microfone, ao invés de ler de uma memória externa, não será necessário efetuar nenhuma mudança no módulo de pré-ênfase. Além disso, o uso de módulos permite uma maior organização do código e facilitará a conexão com a próxima etapa a ser implementada, que será o janelamento. Por outro lado, esta estratégia implica na chamada de funções (ou métodos) de outra biblioteca, gerando assim um retardo no processamento.

## 6. Análise dos Resultados

Diante das adaptações propostas por [Silva 2006] e que foram descritas na Seção 5, foram realizados alguns testes, de modo a identificar o real ganho de desempenho comparado ao cálculo original. Para isso, foram considerados o número de ciclos de *clock* e o tempo de execução para cada uma das implementações.

Para execução dos testes, foram utilizadas as amostras do sinal de voz referentes à sílaba “ba”. O sinal foi capturado utilizando uma taxa de amostragem de 11 KHz e 16 bits de resolução, totalizando 6.652 amostras, cujos valores variam de -10225 a 7828. Uma vez que, as amostras eram processadas à medida que eram lidas do cartão SD, o tempo total de execução sofria pequenas variações a cada execução. Isso acontece já que, o tempo de leitura do cartão pode sofrer pequenas variações à cada solicitação. Diante deste comportamento, os valores considerados neste trabalho, para o tempo total de execução e número de ciclos de *clock,* foram baseados na média de cinco execuções consecutivas.

Os valores foram obtidos a partir da função *micros()* da biblioteca do Arduino, a qual retorna o tempo de execução desde que o programa foi inicializado. Subtraindo-se o valor retornado por *micros()* em diferentes pontos do programa, é possível medir o tempo de execução de determinados trechos de código do mesmo. Esta estratégia foi utilizada para obter o tempo de execução da pré-ênfase isolada, ou seja, sem considerar o tempo para leitura das amostras no cartão de memória (Figura 4).

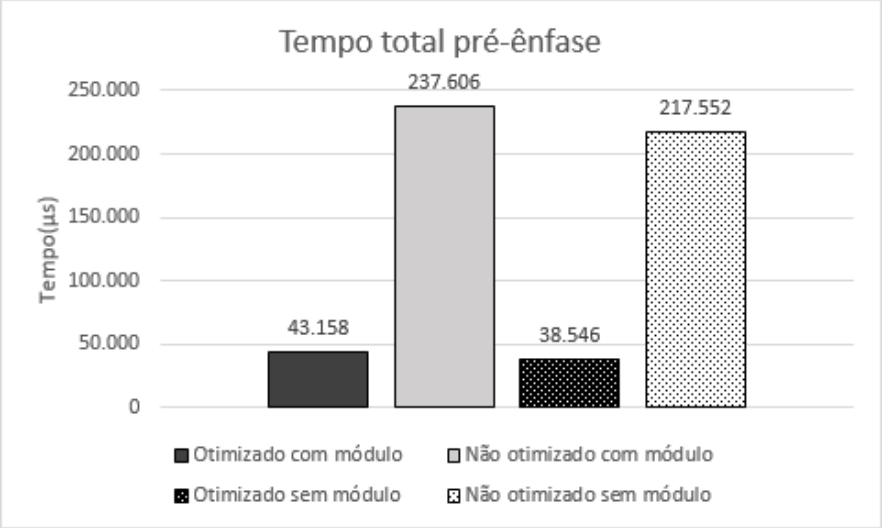


**Figura 4. Tempo de execução em microssegundos para a pré-ênfase.**

Utilizando a mesma função, também foi possível calcular o número total de ciclos de *clock* para executar a pré-ênfase no ATmega328. Uma vez que, o *clock* base do microcontrolador é de 16Mhz, e sendo que 1 Hz equivale a 1 ciclo por segundo, pode-se concluir que o ATmega328 executa 16.000 ciclos por segundo. Assim, com o tempo de execução medido em microssegundos e uma regra de três simples, é possível obter o total de ciclos para cada execução (Figura 5).

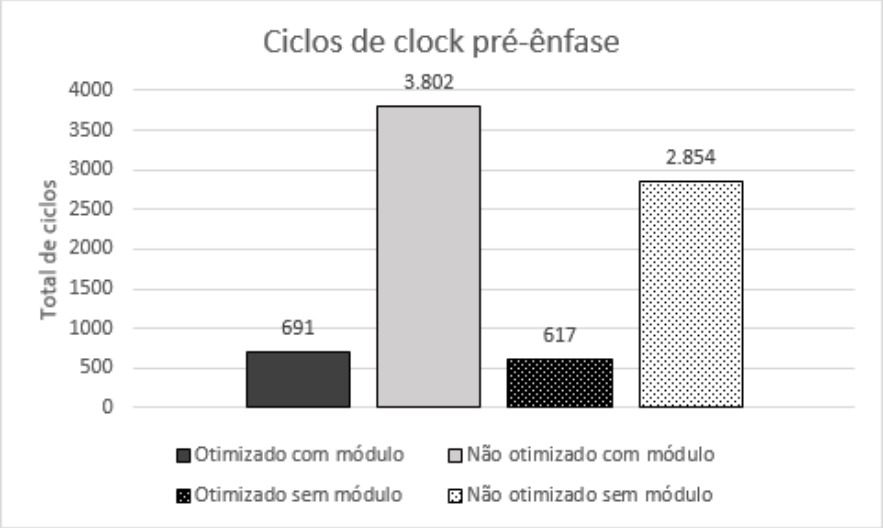


**Figura 5. Total de ciclos de *clock* para a pré-ênfase.**

Analisando os gráficos das Figuras 4 e 5, é possível observar um ganho considerável no desempenho, para a função de pré-ênfase simplificada em relação à implementação tradicional. A redução no tempo e número de ciclos é de praticamente 1/5 do valor inicial. No que se refere ao uso de módulos separados, existe um aumento de aproximadamente 12% para este tipo de implementação, em relação à versão sem módulos.

A decisão de optar ou não pelo uso de módulos vai depender das prioridades do sistema a ser desenvolvido. Caso o tempo de resposta seja o mais importante, talvez valha à pena sacrificar todas as vantagens da modularização. Como exemplo destas, além da que foi citada anteriormente, que é a facilidade de mudar a origem das amostras de voz para um microfone, tem-se a possibilidade de conectar, ao mesmo tempo, a saída da pré-ênfase a diferentes módulos, como o de janelamento e o PWM do Arduino. Ou seja, a decisão de usar, ou não, uma implementação modularizada, consiste de um *tradeoff* entre desacoplamento e tempo de resposta.

## 7. Conclusões e Considerações Finais

O crescente interesse por aplicações de reconhecimento de fala faz com que muitas pesquisas estejam sendo realizadas na área de processamento digital de sinais de voz. No entanto, a implementação deste tipo de aplicação em sistemas embarcados com poucos recursos computacionais torna-se complexa, uma vez que, estas exigem uma certa capacidade de armazenamento e processamento.

O objetivo deste trabalho foi analisar o impacto no desempenho, diante da implementação de simplificações matemáticas em uma etapa típica de um sistema de reconhecimento de fala. Para isso, foram considerados o tempo e número de ciclos de *clock,* para execução da pré-ênfase de voz. Embora esta seja uma etapa relativamente simples, diante de resultados tão positivos, como a redução para 1/5 do tempo original, acredita-se que, utilizando estratégias semelhantes nas demais etapas do processo, será possível implementar um sistema de reconhecimento em um dispositivo com recursos

de *hardware* limitados. A próxima etapa deste trabalho irá focar na função de divisão em quadros e janelamento.

## Referências

BENZEGHIBA, M. F. and BOULARD, H. On the combination of speech and speaker recognition. IDIAP-RR 19, IDIAP – Dalle Molle Institute for Perceptual Artificial Intelligence, 2003.

DA CUNHA, A. M. and VELHO, L. Métodos Probabilísticos para Reconhecimento de Voz. Laboratório VISGRAF – Instituto de Matemática Pura e Aplicada, 2003, 62p. Relatório Técnico, 2003.

DE LIMA, A. A., FRANCISCO, M. S., NETTO, S.L., and RESENDE JR., G. V. Análise Comparativa de Parâmetros em Sistemas de Reconhecimento de Voz. In: XVIII SIMPÓSIO BRASILEIRO DE TELECOMUNICAÇÕES, 2000.

FURUI, S. *Cepstral Analysis Technique for Automatic Speaker Verification*. IEEE Transactions on Acoustics, Speech and Signal Processing Magazine, v. 29, 1981.

J. CIPRIANO, Desenvolvimento de Arquitetura Para Sistemas de Reconhecimento Automático de Voz Baseados em Modelos Ocultos de Markov. 123 f . PhD thesis, Tese (Doutorado em Ciência da Computação)_Universidade Federal do Rio Grande do Sul, Porto Alegre, 2001.

KLEIJN, W. B. and PALIWAL, K. K. Speech Coding and Synthesis, Elsevier Science. B. V., 1998.

PETRY, A., ZANUZ, A. e BARONE, D. A. C. Reconhecimento Automático de Pessoas Pela Voz Através de Técnicas de Processamento Digital de Sinais. SEMAC 2000.

RABINER, L. R.; SCHAFER, R. W. *Digital processing of speech signals. New Jersey: Prentice Hall,* 1978.

SHAUGHNESSY, D. O. *Speech Communications, Human and machine*. New York: IEEE Press, 2000.

SILVA, D. D. C. Desenvolvimento de um ipcore de pré-processamento digital de sinais de voz para aplicação em sistemas embutidos. Dissertação de Mestrado – Universidade federal de campina grande, 2006.

SILVA, D. D. C.. Reconhecimento de Fala Contínua para o Português Brasileiro em Sistemas Embarcados. 198p. Tese (Doutorado em Ciências no Domínio da Engenharia Elétrica). Centro de Engenharia Elétrica e Informática, Universidade Federal de Campina Grande, Campina Grande, 2011.

MCROBERTS, Michael. Arduino Básico, Novatec, São Paulo, 2011