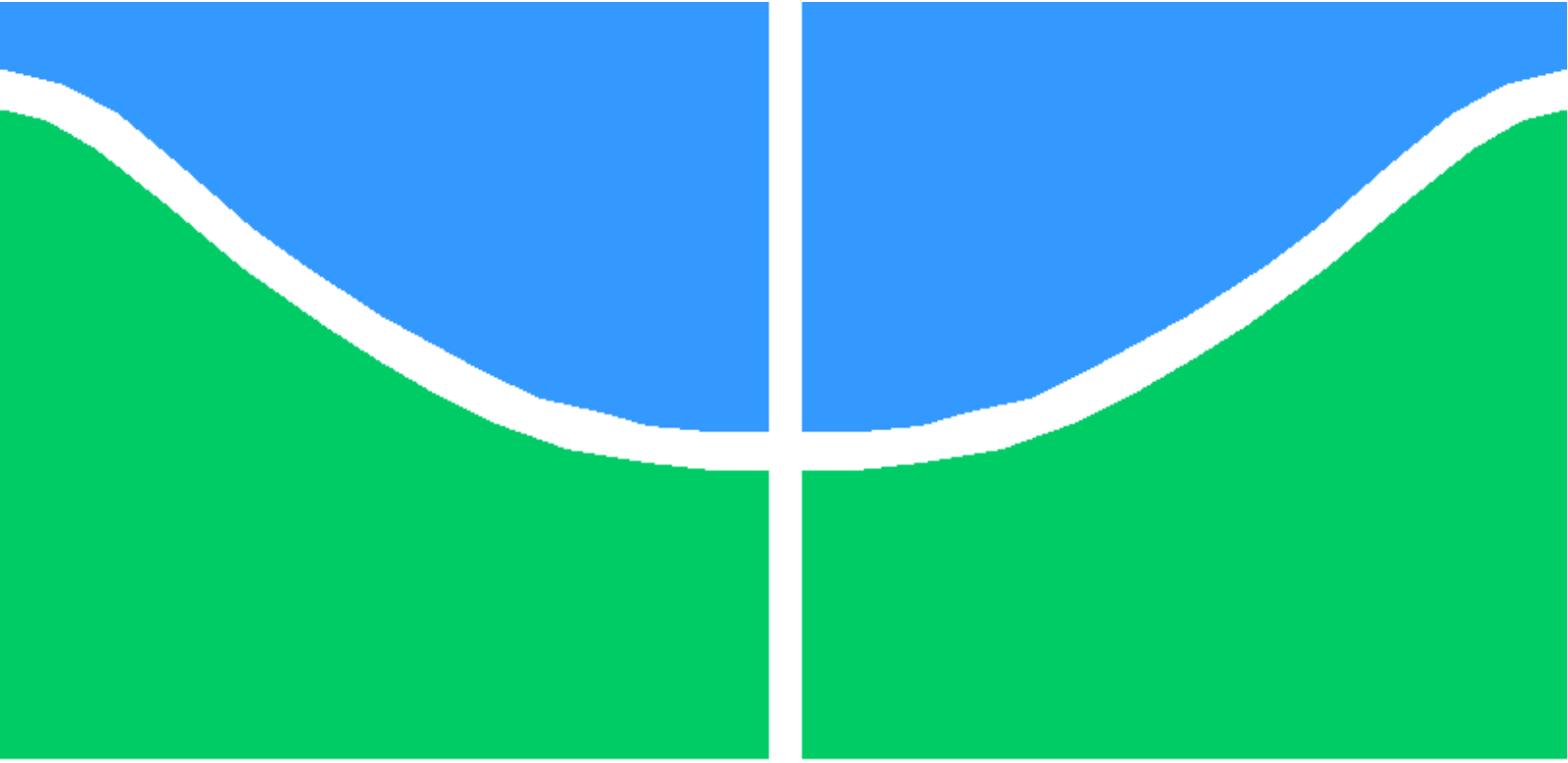

TRABALHO DE GRADUAÇÃO

ANÁLISE DE CONSUMO DE ENERGIA E PROCESSAMENTO DE SINAIS NA PLATAFORMA *MICAZ*

Sávio Oliveira de Almeida Neves

Brasília, julho de 2014

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASILIA
Faculdade de Tecnologia

TRABALHO DE GRADUAÇÃO

ANÁLISE DE CONSUMO DE ENERGIA E PROCESSAMENTO DE SINAIS NA PLATAFORMA *MICAZ*

Sávio Oliveira de Almeida Neves

Relatório submetido ao Departamento de Engenharia Elétrica como requisito parcial para obtenção do grau de Engenheiro de Redes de Comunicação

Banca Examinadora

Prof. Renato Mariz de Moraes, ENE/UnB _____
Orientador

Prof. João Luiz Azevedo de Carvalho, ENE/UnB _____
Examinador interno

Prof. Marcelo Menezes de Carvalho, ENE/UnB _____
Examinador interno

Dedicatória

Ao meu tio Claudio (in memorian), ao meu avô Mario (in memorian) e à minha avó Theodorina (in memorian), pois estarão felizes com a minha conquista onde se encontrarem.

Sávio Oliveira de Almeida Neves

Agradecimentos

Gostaria de agradecer primeiramente a Deus por me dar forças para aguentar essa jornada de muito trabalho e aprendizado, de momentos difíceis e de muitas alegrias.

Agradeço ao meu pai Swami Neves por ser meu exemplo de caráter, hombridade e perserverança. Agradeço à minha mãe Marisa Neves por todo carinho e amor, que diversas vezes cedeu seus sonhos para me fornecer o melhor. Agradeço ao meu irmão Swami Filho pelo companheirismo, amizade e parceria. Agradeço à minha avó Pura pelas orações. Obrigado por ser a melhor família deste mundo e sempre estar presente nas inúmeras vezes de que precisei, fosse para desabafar as angústias da vida ou simplesmente ouvir minhas conversas de engenheiro, mesmo sem entender nada.

Agradeço humildemente ao Professor Renato Mariz, por sua calma e organização, conseguindo pôr em ordem as minhas ideias. Professor, muito obrigado por realmente saber o significado da palavra orientar.

Agradeço aos Professores Marcelo Menezes, Ugo Dias, Darli Mello, André Noll e Paulo Portela, por todos os ensinamentos e por conseguirem tirar o melhor deste aluno. Agradeço também ao João Paulo Leite, muito obrigado por seu auxílio e paciência nas inúmeras vezes em que entrei no LEMON em busca de ajuda. E um agradecimento especial aos professores Marcus Lamar e João Luiz pelas diversas dicas durante a execução deste trabalho.

Agradeço aos amigos Diguin e Éverton pelos bons momentos na Telebrás, derrubando servidores ou cantando um Bob Marley para animar o ambiente de trabalho. Agradeço também ao amigo Evandro, companheiro de inúmeras horas, inúmeras mesmo, no GPDS, seja trocando informações sobre os projetos ou conversando sobre aleatoriedades da vida.

Agradeço ao meu amigo João Eduardo pela amizade e pelo apoio na minha recuperação cirúrgica. Agradeço ao pessoal da natação e ao Nando do cavaquinho, com certeza esses momentos de distração me deram forças para continuar seguindo em frente.

E por fim, obrigado Manu, por todo o carinho, por me ouvir, por me aconselhar, pelos cinemas, pelo sushi, pelos teatros e pelas conversas, relacionadas aos trabalhos ou extremamente inúteis e sem sentido algum.

Obrigado a todos que estiveram ao meu lado durante esta caminhada.

Hakuna Matata
É lindo dizer
Hakuna matata, sim vai entender
Os seus problemas você deve esquecer, isso é viver
é aprender hakuna matata

Sávio Oliveira de Almeida Neves

RESUMO

Este trabalho investiga o consumo de energia dos *motes MICAZ* presentes nas redes de sensores. Inicialmente, o trabalho realiza a estimação do gasto de energia para aquisição de dados, processamento e transmissão. É utilizado um sinal eletrocardiográfico a fim de realizar a estimação desses gastos, considerando que esse sinal passa pelos seguintes processamentos: transformada discreta de cosseno seguido do cálculo da variância dos componentes e *Compressed Sensing*. Esses processamentos são utilizados a fim de reduzir o montante de dados a serem transmitidos e, consequentemente, reduzir o gasto de transmissão. Através das estimações realizadas, nota-se que a melhor estratégia para reduzir o consumo de transmissão e que ocasiona um menor aumento no processamento seria uma estratégia baseada somente na transformada discreta de cosseno. A partir dessa verificação, são implementadas quatro aplicações na plataforma *MICAZ*, duas que utilizam a transformada de cosseno e duas que não a utilizam. As aplicações coletam dados sobre a temperatura ambiente e luminosidade ambiente. Para o caso das aplicações que utilizam a transformada de cosseno, caso os coeficientes sejam nulos, eles não seriam transmitidos, reduzindo assim o consumo de energia da transmissão. É verificado uma redução do consumo de energia quando se utiliza a transformada em relação aos casos em que a transformada não é utilizada. Além do mais, ao comparar os sinais recebidos que utilizam a transformada de cosseno com os sinais que não utilizam a transformada, verifica-se um erro quadrático médio pequeno, validando as aplicações que adotam a transformada.

ABSTRACT

This study investigates the energy consumption of the motes *MICAZ* present in sensor networks. Initially, the work accomplishes the estimation of energy expenditure for data acquisition, processing and transmission. An electrocardiographic signal is used to perform the estimation of these costs, whereas this signal goes through the following processes: DCT followed by the calculation of the variance of components and Compressed Sensing. These processes are used to reduce the amount of data to be transmitted and thus reducing the cost of transmission. Through the estimations performed, it is noted that the best strategy to reduce the consumption of transmission and that causes a smaller increase in processing would be a strategy based only on the DCT. From this verification, four applications are implemented in *MICAZ* platform, two using the cosine transform and two which do not use it. The applications collect data of the environment temperature and environment light. For the case of applications that use cosine transform, the coefficients equal to zero are not transmitted, reducing the consumption of transmission. A reduction in energy consumption is observed when using the transform in relation to cases where the transform is not used. Furthermore, comparing the received signals using the cosine transform with signals that do not use the transform, it turns a small mean square error, validating applications that adopt the use of the transform.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SÍMBOLOS

Símbolos Latinos

C	Capacitância	[F]
d	Distância	[m]
E	Energia	[J]
I	Corrente	[A]
T	Tempo	[s]
V	Tensão	[V]

Símbolos Gregos

α	Coeficiente de Perda de Percurso	
ϵ	Constante do Amplificador do Rádio	[pJ/bit/m^2]
δ_k	Constante da Condição RIP	
γ	Fator de Utilização	
Φ	*Sensing Matrix*	

Grupos Adimensionais

b	Bits
f	Frequência de Processamento
L	Janela da Técnica *Compressed Sensing* Não Uniforme
n	Constante do Microcontrolador
N	Número de Ciclos para Execução do Programa

Subscritos

A	Ativo
$amost$	Amostra
amp	Amplificador
avg	*Average*
cyc	Ciclos
$elec$	Circuito Eletrônico do Rádio
I	Inativo
i	Nó i
$inst$	Instante
j	Nó j
mem	Memória
o	*Leakage*
pro	Processamento
$read$	Leitura da Memória
Rx	Recepção
$sens$	Aquisição de Dados
sup	Fornecido
T	Térmico
tot	Total
Tx	Transmissão
$write$	Escrita na Memória

Sobrescritos

Siglas

BAN	*Body Area Network*
BPSK	*Binary Phase Shift Keying*
CSMA/CA	*Carrier Sense Multiple Access with Collision Avoidance*
DCT	*Discrete Cosine Transform*
DFT	*Discrete Fourier Transform*
FFD	*Full Function Device*
FFT	*Fast Fourier Transform*
FIR	*Finite Impulse Response*
GPIB	*General Purpose Interface Bus*
GTS	*Guaranteed Time Slot*
IEEE	*Institute of Electrical and Electronics Engineers*
LAN	*Local Area Network*
MAC	*Medium Access Control*
MPDU	*Medium Access Control Layer Data Unit*
MSDU	*Medium Access Control Layer Service Data Unit*
PAN	*Personal Area Network*
PHY	*Physical*
PPDU	*Physical Layer Protocol Data Unit*
PSDU	*Physical Layer Service Data Unit*
RFD	*Reduced Function Device*
RIP	*Restricted Isometry Property*
USB	*Universal Serial Bus*

Capítulo 1

Introdução

Este capítulo apresenta os aspectos principais do trabalho, contextualizando o problema proposto no âmbito das redes de sensores. Além disso, a motivação e as contribuições são destacadas, e a estrutura de todo o manuscrito é apresentada.

1.1 Contextualização

Redes de sensores formam uma sub-classe de redes *ad hoc* sem fio, sendo caracterizada pela presença de nós coletores de informação, conhecidos como *Leaf Node*, e a presença de um nó que receberá essa informação e a repassará para alguma rede local a fim de distribuir essa informação aos usuários (esse nó é conhecido como *Sink Node*). Esses nós, que são compostos por um *transceiver* de rádio, microcontrolador e sensores aquisitores de dados, também podem ser denominados de *motes*. Por ser uma sub-classe de rede *ad hoc*, grande parte das características são equivalentes, como por exemplo a falta de infraestrutura, utilização de saltos entre os nós para enviar a informação, a construção de tabelas de roteamento ocorre de maneira distribuída entre os nós, evitando assim a dependência de um nó de gerenciamento central, entre outras características. Ao se trabalhar com redes de sensores, há a possibilidade de realizar diversas aplicações, entre elas pode-se citar: monitoramento de florestas, monitoramento de pessoas e monitoramento de animais.

Levando em consideração a diversidade de aplicações existentes, surge-se a possibilidade de verificar a capacidade dos *motes* realizarem algum tipo de processamento nos sinais que estão sendo adquiridos, seja um processamento individual ou um processamento distribuído. Entretanto, os *motes* são equipamentos que possuem restrições no processamento, memória e capacidade de energia, portanto é necessário realizar um estudo de como implementar tais sistemas de modo a aumentar a sua vida útil. Levando todos esses aspectos em consideração, este projeto de graduação visa verificar a aplicabilidade da realização de processamento digital de sinais em *motes MICAZ* [1], que é um *mote* produzido pela *Crossbow*.

1.2 Definição do problema

Esta monografia trata o estudo da eficiência de energia em redes de sensores, verificando o potencial de processamento de *motes MICAZ*. Há o interesse de inferir a possibilidade de realização de técnicas de processamento de sinais a fim de reduzir o montante de dados a serem transmitidos, explorando a redundância temporal da informação. Este interesse justifica-se pois grande parte da literatura indica que o fator preponderante no consumo de energia dos *motes* é o gasto de transmissão, sendo pouco estudado a possibilidade de realizar processamentos locais para reduzir o montante de dados, uma vez que os *motes* possuem diversas restrições na parte de processamento, memória e capacidade de geração de energia. Trabalhos futuros podem estender esta ideia ao explorar a redundância espacial da informação, no caso, a redundância presente entre as informações obtidas pelos diversos *motes* existentes na rede de sensores.

1.3 Contribuições do trabalho

As contribuições deste trabalho são:

- Proposta para consumo de energia para transmissão de dados na plataforma *MICAZ*;
- Cálculo do gasto de energia para *motes MICAZ*, levando em consideração a aquisição de um sinal eletrocardiográfico, processamento do sinal e transmissão dos dados;
- Implementação de aplicações que coletam informações sobre a temperatura e luminosidade ambiente na plataforma *MICAZ*;
- Implementação de algoritmo rápido para o cálculo da transformada discreta de cosseno na plataforma *MICAZ*;
- Avaliação do consumo de energia para as aplicações em que há o cálculo da transformada e para as aplicações em que não há o cálculo da transformada;
- Comparação do sinal recebido quando há o cálculo da transformada e quando não há o cálculo da transformada.

1.4 Apresentação do manuscrito

O restante desta monografia está organizada da seguinte forma. No Capítulo 2 são explicados os modelos de energia presentes na liteturatura, tendo um enfoque maior para o modelo que foi utilizado para as estimações, e também há uma explicação sobre as técnicas de processamento de sinais que foram utilizadas no decorrer do trabalho. Além disso, há uma rápida revisão sobre conceitos básicos sobre a norma IEEE 802.15.4, sobre redes de sensores e sobre o *TinyOS* e *NesC*, sendo essa a linguagem adotada para programar os *motes*, visando o seu melhor desempenho baseado no gasto de energia. Em seguida, o Capítulo 3 descreve a estimação da energia utilizada, ilustrando

detalhadamente o consumo para aquisição de dados, processamento e transmissão. Também há a explicação da fórmula proposta para o fator de utilização do *transceiver* de rádio baseado na norma IEEE 802.15.4 e no *Mote MICAZ*. E além disso, é estruturada a realização da implementação em *TinyOS* do sistema proposto. Os resultados obtidos das estimações e da parte prática e seus pseudo-códigos estão presentes no Capítulo 4, seguidos das conclusões no Capítulo 5. Os anexos contêm material complementar.

Capítulo 2

Revisão Bibliográfica

Neste capítulo, são apresentadas as definições e conceitos necessários para a elaboração deste trabalho.

2.1 Introdução

Uma das principais preocupações para implementação de uma rede de sensores é aumentar a vida útil dos nós, já que uma possível substituição de determinado nó pode ser problemática ou impraticável. Entre os modelos existentes, pode-se citar [2], onde há a descrição de modelos de consumo de energia para a transmissão e recepção de dados, além de modelos para o processamento dos nós. Já em [3], há o desenvolvimento de fórmulas para o gasto de transmissão e recepção de dados, processamento, aquisição de dados e escrita e leitura de dados na memória. Além disso, também ocorre a descrição do gasto de energia para os nós que coordenam a rede ou um *cluster* da rede. Todavia, a grande maioria dos trabalhos existentes realizam o desenvolvimento dos modelos por simulações, e dificilmente há implementações para comprovar tais modelos. Nas simulações realizadas nesses trabalhos, há a indicação de que a parte de comunicação, recepção e transmissão de dados, é a responsável pelo maior consumo de energia nos *motes*.

Por outro lado, há a possibilidade de realizar processamentos na informação a ser transmitida, a fim de reduzir o montante de dados e o consumo por comunicação no sensor. Entre os trabalhos que exploram essa ideia, pode-se citar [4], onde é verificado a possibilidade de utilizar a transformada de cosseno para aproveitar a redundância espacial e temporal dos dados a fim de reduzir o montante de dados a ser utilizado, porém a implementação realizada utiliza-se de poucos coeficientes no cálculo e não é verificado o gasto de energia dos nós. Outro trabalho relacionado é [5], onde foi implementada uma codificação de vídeo distribuída que foi comparada com o desempenho de uma transformada de cosseno. Todavia, a codificação foi implementada em uma plataforma adicional, e não no microcontrolador do *mote*, além de verificar o gasto baseado somente no decaimento da tensão das pilhas.

A seguir serão explicados os modelos empregados em [2] e [3]. Além de explicar alguns conceitos de [4] e [5], a fim de possibilitar um bom entendimento deste texto.

2.2 Norma IEEE 802.15.4 e Redes de Sensores

A norma IEEE 802.15.4 é uma padronização elaborada para nós com baixa taxa de transmissão, baixa complexidade e baixo consumo de energia, sendo realizada para as camadas PHY (do inglês *Physical*) e MAC (do inglês *Medium Access Control*). A taxa de transmissão pode alcançar valores de até 250kbps e o alcance da transmissão é de 10 a 20 metros, normalmente. As redes de sensores podem utilizar as especificaçãoes presentes nessa norma, na Figura 2.1 está ilustrado uma rede de sensores.

Os elementos da rede podem ser classificados da seguinte maneira:

- *Full Function Device* (FFD): são os elementos responsáveis pelas coletas de dados e roteamento de pacotes;
- *Reduced Function Device* (RFD): são os elementos responsáveis pelas coletas de dados;
- *PAN* (do inglês *Personal Area Network*) *Coordinator*: é o elemento responsável pela associação e desassociação de novos elementos da rede.

Algumas aplicações que podem utilizar essa norma:

- Automação predial;
- Redes de sensores;
- Periféricos de computador;
- Jogos;
- *Smart energy*;
- *Health care*.

A camada PHY para essa norma é responsável por:

- Ativação e desativação do *transceiver* de rádio;
- Detecção de energia;
- Indicação de qualidade do enlace;
- Avaliação de canal livre;
- Seleção do canal de frequência;
- Recepção e transmissão de dados.

A camada MAC para essa norma é responsável por:

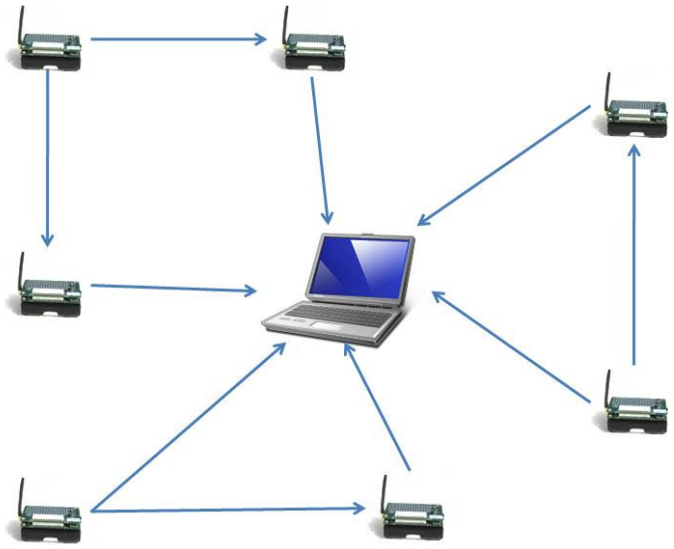

Figura 2.1: Rede de Sensores.

- Gerar *beacons* se o equipamento for coordenador;
- Sincronizar os *beacons*;
- Suporte para associação e desassociação na rede;
- Utilização do protocolo CSMA/CA (do inglês *Carrier Sense Multiple Access with Collision Avoidance*);
- Manutenção e tratamento de slots GTS (do inglês *Guaranteed Time Slot*);
- Prover enlace confiável entre dois pares de entidade MAC.

O pacote da camada PHY é subdividido conforme ilustrado na Figura 2.2.

O preâmbulo tem tamanho de 32 bits, os quais são utilizados para sincronização na recepção do pacote. O delimitador possui 8 bits, sendo responsável por indicar o fim do preâmbulo. O cabeçalho PHY indica o comprimento do PSDU (do inglês *Physical Layer Service Data Unit*), possuindo também 8 bits. E por fim, o PSDU possui até 127 bytes de informação.

Há 27 canais disponíveis para a transmissão dos dados, sendo as características sintetizadas na Tabela 2.1.

O pacote da camada MAC é subdividido conforme ilustrado na Figura 2.3.

O controle de quadro tem tamanho de 2 bytes, o número de sequência tem 1 byte e a informação de endereço pode possuir até 20 bytes, esses três campos formam o cabeçalho do quadro MAC. O MSDU (do inglês *Medium Access Control Layer Service Data Unit*) possui tamanho variável e é

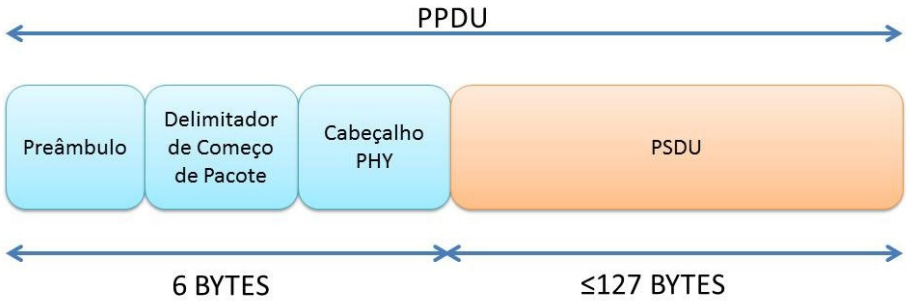

Figura 2.2: Pacote da Camada PHY.

Tabela 2.1: Características Modulação.

PHY	Banda de Frequência	Taxa de Bit(kb/s)	Taxa de Símbolo(kbaud)	Modulação
868/915	868,0-868,6 MHz	20	20	BPSK
MHz PHY	902,0-928,0 MHz	40	40	BPSK
2,4 GHz PHY	2,4-2,4835 GHz	250	62,5	16-ary ortogonal

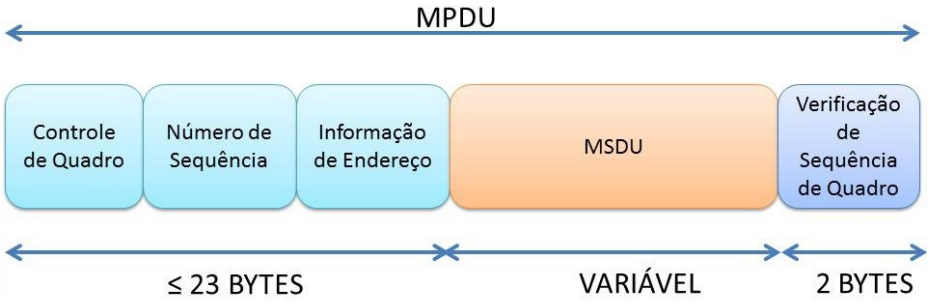

Figura 2.3: Pacote da Camada MAC.

responsável pelos bits de informação das camadas superiores. E por fim, a verificação de sequência de quadro possui 2 bytes.

2.3 Modelos de Consumo de Energia

Foram estudados diversos modelos de consumo de energia. Porém, os modelos que melhor se adequaram a pesquisa foram os modelos das referências [2] e [3].

Para o modelo da referência [2], há as seguintes fórmulas para o consumo de energia pelos circuitos de comunicação:

$$E_{Tx}(b,d) = E_{elec}.b + \epsilon_{amp}.b.d^2 \tag{2.1}$$

e

$$E_{Rx}(b) = E_{elec}.b. \tag{2.2}$$

Nessas fórmulas, E_{Tx} é a energia em joules para transmissão, E_{Rx} é a energia em joules para recepção, b é a quantidade de bits a serem transmitidos, d é a distância em metros entre transmissor e receptor, E_{elect} é a energia dissipada em joules pelo rádio para a transmissão de um bit e ϵ_{amp} é uma constante que relaciona a energia gasta do amplificador para garantir uma relação sinal ruído desejada. Essa constante é dada em joules por bit por metros ao quadrado.

Para o consumo de energia referente ao processamento, tem-se

$$E_{pro} = C_{tot}.V_{sup}^2 + V_{sup}.(I_o.e^{V_{sup}/(nV_T)}).(N/f). \tag{2.3}$$

Nessa fórmula, E_{pro} é a energia para processamento em joules, C_{tot} é a capacitância total trocada pelo programa em farad, V_{sup} é a tensão de alimentação do circuito em volts, I_o é a corrente *leakage* em ampère, n é uma constante que depende do microcontrolador, V_T é a tensão térmica em volts, N é o número de ciclos que o programa utiliza para executar e f é a frequência de processamento em hertz.

Já para o modelo da referência [3], há a seguinte fórmula para a aquisição de dados do sensor:

$$E_{sens}(b) = b.V_{sup}.I_{sens}.T_{sens}. \tag{2.4}$$

Nessa fórmula, E_{sens} é a energia para a aquisição de dados em joules, b é a quantidade de bits processados, V_{sup} é a tensão de alimentação do circuito em volts, I_{sens} é a corrente requerida para a atividade de aquisição em ampère e T_{sens} é o tempo total para aquisição em segundos.

Para leitura e escrita na memória do sensor tem-se

$$E_{mem}(b) = \frac{b.V_{sup}}{8}.(I_{write}.T_{write} + I_{read}.T_{read}). \tag{2.5}$$

Nessa fórmula, E_{mem} é a energia gasta para as operações na memória em joules, b é a quantidade de bits processados, V_{sup} é a tensão de alimentação do circuito em volts, I_{write} é a corrente requerida para operações de escrita na memória em ampère, T_{write} é o tempo total para escrita na memória em segundos, I_{read} é a corrente requerida para operações de leitura da memória em ampère e T_{read} é o tempo total para leitura da memória em segundos.

Para o processamento tem-se

$$E_{pro}(b, N_{cyc}) = b.N_{cyc}.C_{avg}.V_{sup}^2 + b.V_{sup}.(I_o.e^{V_{sup}/(n.V_T)}).(N_{cyc}/f). \tag{2.6}$$

Nessa fórmula, E_{pro} é a energia consumida pelo processamento em joules, b é a quantidade de bits processados, N_{cyc} é o número de ciclos do *clock* por tarefa, C_{avg} é a capacitância média

trocada por ciclo em farad, V_{sup} é a tensão de alimentação do circuito em volts, I_o é a corrente *leakage* em ampère, n é uma constante que depende do microcontrolador, V_T é a tensão térmica em volts e f é a frequência de processamento em hertz.

Para comunicação tem-se

$$E_{Tx}(b, d_{ij}) = b.E_{elec} + b.d_{ij}^{\alpha}.E_{amp} \tag{2.7}$$

e

$$E_{Rx}(b) = b.E_{elec}. \tag{2.8}$$

Nessas fórmulas, E_{Tx} é a energia consumida para transmissão em joules, E_{Rx} é a energia consumida para recepção em joules, b é a quantidade de bits processados, d_{ij} é a distância de transmissão em metros, E_{elec} é a energia dissipada em joules pela parte eletrônica do rádio para transmissão e recepção de um bit, α é o coficiente de perda de percurso e E_{amp} é a energia dissipada pelo amplificador de potência em joules por bit por metros elevados ao valor de α.

No artigo, também foi relacionado o consumo devido a utilização do sensor, ou seja, todas as fórmulas supracitadas foram multiplicadas por uma constante, γ, que é equivalente ao fator de utilização em uma unidade de tempo. E também foi considerado o consumo em modo inativo do sensor, onde o complementar do fator de utilização multiplicava a energia consumida em modo inativo. Por fim, soma-se os valores de energia consumida em modo ativo e em modo inativo para se obter o consumo total.

Além do mais, no artigo também há o cálculo de consumo de energia de *cluster heads* na rede de sensores, porém, como o trabalho desenvolvido aqui realizou processamento em somente um sensor, esses aspectos não foram consideradados até o momento.

É interessante notar que o cálculo do gasto de transmissão nos dois trabalhos utiliza a perda de percurso e a distância entre o transmissor e receptor. Todavia, nesta monografia é utilizada outra fórmula para o gasto de transmissão, explicada no Capítulo 3, que não utiliza esses dois fatores.

Há a possibilidade de utilizar esses dois fatores com o propósito de verificar a quantidade média de retransmissões necessárias para o envio de dados de maneira confiável em ambientes generalizados, utilizando também desvanecimento em pequena e larga escala. Ao obter essa quantidade média de retransmissões, multiplica-se esse número médio com o valor obtido para o fator de utilização do *transceiver* de rádio, assim calcula-se o gasto de energia para uma aplicação em que a transmissão ocorre de maneira confiável. Entretanto, ao utilizar as fórmulas apresentadas em [2] e [3] acarretaria uma superestimação do gasto de transmissão.

Para maiores informações, consultar as referências [2] e [3].

2.4 *TinyOS* e *NesC*

O *TinyOS* é um sistema operacional criado para redes de sensores e tem como foco o baixo consumo de energia, sendo utilizado para os microcontroladores de *motes*. O *TinyOS* possibilita a construção de aplicações mais facilmente, uma vez que provê uma série de abstrações e serviços, entre eles pode-se citar: aquisição de dados, comunicação, armazenamento e temporizadores. Além disso, o *TinyOS* roda em dúzias de plataformas genéricas, das quais a maioria suporta a adição de novos sensores.

O *NesC* por sua vez é uma vertente da linguagem C que possui características a fim de diminuir o tamanho do código, evitando assim *bugs* em baixo nível. As aplicações, sistemas e o próprio *TinyOS* são escritos com o *NesC*.

Os conceitos básicos do *NesC* são os seguintes:

- Separação de construção e composição: programas são construídos baseados em componentes, que são interligados. Componentes definem dois escopos, um para sua especificação (contém o nome de suas interfaces) e outro para sua implementação;
- A especificação do comportamento de um componente é dado em termos do conjunto de interfaces. As interfaces podem ser usadas ou providas pelo componente, sendo representadas pela funcionalidade necessária para realização de uma tarefa ou a funcionalidade provida para o usuário, respectivamente;
- Interfaces são bidirecionais: elas especificam o conjunto de funcionalidades a serem implementadas pelo provedor da interface (comandos) e o conjunto a ser implementado pelo usuário da interface (eventos);
- Componentes estão estaticamente ligados entre si através de suas interfaces.

Para maiores informações, consultar referência [6].

2.5 Transformada Discreta de Cosseno

A Transformada Discreta de Cosseno foi estudada a fim de implementá-la nos *motes MICAZ*, uma vez que esta é usada em diversos padrões de compressões existentes. A transformada X_k da série x_n pode ser calculada a partir do seguinte somatório

$$X_k = c_k \sum_{n=0}^{M-1} x_n \cos[\frac{(n+0.5)k\pi}{M}], \tag{2.9}$$

em que

$$c_k = \begin{cases} \frac{1}{\sqrt{2}} & \text{se } k = 0, \\ 1 & \text{caso contrário.} \end{cases}$$

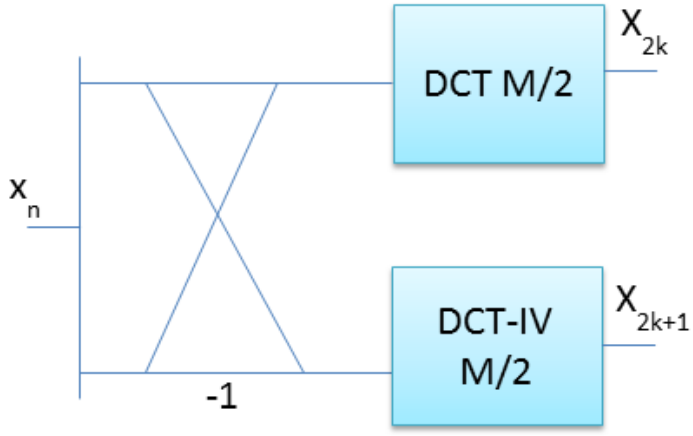

Figura 2.4: Diagrama de Blocos DCT.

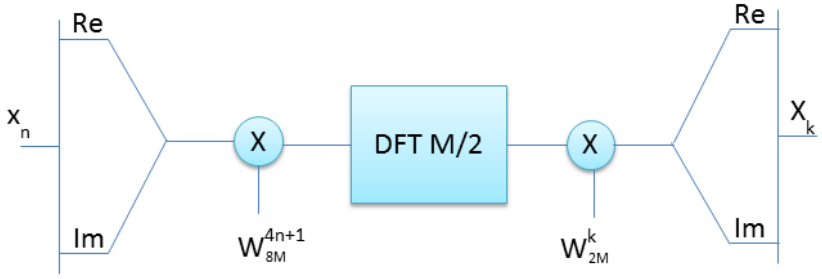

Figura 2.5: Diagrama de Blocos DCT-IV.

e M é a quantidade de elementos pertencentes a série x_n.

2.5.1 Algoritmo Rápido

Neste trabalho foi utilizado um algoritmo rápido para o cálculo da DCT (do inglês *Discrete Cosine Transform*), esse algoritmo é explicado no livro [7]. O algoritmo para o cálculo da DCT é baseado no algoritmo da DCT-IV, e esse, por sua vez, usa o algoritmo *Split Radix FFT* (do inglês *Fast Fourier Transform*). Então, nesta subseção é realizada uma descrição rápida de ambos. Nas Figuras 2.4, 2.5 e 2.6, são ilustrados diagramas de blocos que representam a organização do cálculo da DCT, DCT-IV e *Split Radix FFT*.

Conforme pode ser visto na Figura 2.4, este algoritmo funciona de maneira recursiva, calculando a DCT com uma quantidade menor de elementos através da DCT-IV. O algoritmo da DCT-IV possui os seguintes passos:

1. Dado um vetor com M números reais, x_n, faça o seguinte ordenamento para n=0,...,$\frac{M}{2} - 1$:

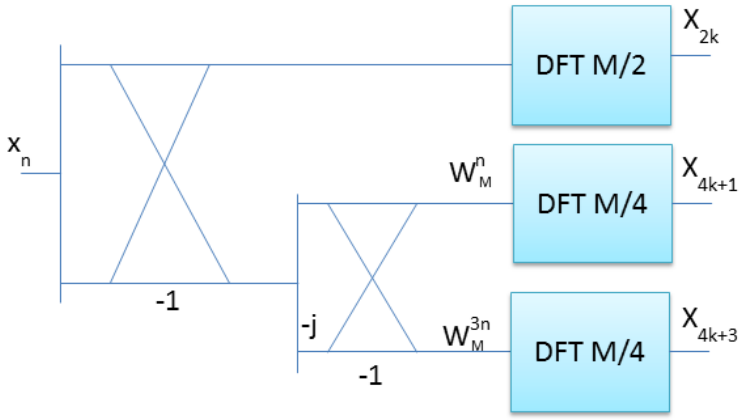

Figura 2.6: Diagrama de Blocos *Split Radix FFT*.

(a) $x_n \leftarrow x_{2n}$

(b) $x_{n+\frac{M}{2}} \leftarrow x_{M-1-2n}$

2. Calcular para n=0,...,$\frac{M}{2}-1$:

(a) $x_n \leftarrow \mathrm{Re}\{[x_n + jx_{n+\frac{M}{2}}]\exp[-j(n+\frac{1}{4})\frac{\pi}{M}]\}$

(b) $x_{n+\frac{M}{2}} \leftarrow \mathrm{Im}\{[x_n + jx_{n+\frac{M}{2}}]\exp[-j(n+\frac{1}{4})\frac{\pi}{M}]\}$

3. Calcular:

(a) $[x_0...x_{M-1}] \leftarrow \mathrm{DFT}\{[x_0...x_{M-1}], \frac{M}{2}\}$

4. Calcular:

(a) $x_n \leftarrow x_n \exp(-j\frac{n\pi}{M}), n = 1, 2, ..., \frac{M}{4}-1, \frac{M}{4}+1, ..., \frac{M}{2}-1$

(b) $x_{\frac{M}{4}} \leftarrow \sqrt{2}(1-j)x_{\frac{M}{4}}$

5. Reordenar:

(a) $x_{2n} \leftarrow x_n, n = 0, 1, ..., \frac{M}{2}-1$

(b) $x_{M-1-2n} \leftarrow -x_{n+\frac{M}{2}}, n = 0, 1, ..., \frac{M}{2}-1$

No passo 3, a DFT (do inglês *Discrete Fourier Transform*) é calculada pelo algoritmo *Split Radix FFT* que é descrito a seguir:

1. Dado um vetor com M números complexos, x_n, calcular para n=0,...,$\frac{M}{2}-1$:

(a) $x_n \leftarrow x_n + x_{n+\frac{M}{2}}$

(b) $x_{n+\frac{M}{2}} \leftarrow x_n - x_{n+\frac{M}{2}}$

2. Calcular para n=0,...,$\frac{M}{4}-1$:

(a) $x_{n+\frac{M}{2}} \leftarrow x_{n+\frac{M}{2}} + jx_{n+\frac{3M}{4}}$

(b) $x_{n+\frac{3M}{4}} \leftarrow x_{n+\frac{M}{2}} - jx_{n+\frac{3M}{4}}$

3. Calcular para n=1,...,$\frac{M}{4} - 1$:

(a) $x_{n+\frac{M}{2}} \leftarrow x_{n+\frac{M}{2}} W_M^n$ para $n \neq \frac{M}{8}$, sendo $W_M = \exp(-j\frac{2\pi}{M})$

(b) $x_{\frac{5M}{8}} \leftarrow \sqrt{\frac{1}{2}}(1-j)x_{\frac{5M}{8}}$

4. Calcular para n=1,...,$\frac{M}{4} - 1$:

(a) $x_{n+\frac{3M}{4}} \leftarrow x_{n+\frac{3M}{4}} W_M^{3n}$ para $n \neq \frac{M}{8}$, sendo $W_M = \exp(-j\frac{2\pi}{M})$

(b) $x_{\frac{7M}{8}} \leftarrow -\sqrt{\frac{1}{2}}(1+j)x_{\frac{7M}{8}}$

5. Calcular:

(a) $[x_0...x_{\frac{M}{2}-1}] \leftarrow \text{DFT}\{[x_0...x_{\frac{M}{2}-1}], \frac{M}{2}\}$

(b) $[x_{\frac{M}{2}}...x_{\frac{3M}{4}-1}] \leftarrow \text{DFT}\{[x_{\frac{M}{2}}...x_{\frac{3M}{4}-1}], \frac{M}{4}\}$

(c) $[x_{\frac{3M}{4}}...x_{M-1}] \leftarrow \text{DFT}\{[x_{\frac{3M}{4}}...x_{M-1}], \frac{M}{4}\}$

6. Reordenar:

(a) $x_{2n} \leftarrow x_n, n = 0, 1, ..., \frac{M}{2} - 1$

(b) $x_{4n+1} \leftarrow x_{n+\frac{M}{2}}, n = 0, 1, ..., \frac{M}{4} - 1$

(c) $x_{4n+3} \leftarrow x_{n+\frac{3M}{4}}, n = 0, 1, ..., \frac{M}{4} - 1$

Para maiores informações, consultar a referência [7].

2.6 *Compressed Sensing*

A teoria no desenvolvimento da técnica de *Compressed Sensing* revela que sinais esparsos, sinais que podem ser representados a partir de poucas bases pertencentes a um conjunto de bases de projeção, podem ser reconstruídos com uma quantidade menor de amostras do que a indicada pela taxa de Nyquist. Esta possibilidade torna essa técnica promissora para utilização em redes de sensores, uma vez que reduz o processamento necessário para aquisição de sinais.

A teoria de *Compressed Sensing* é resumida a seguir. Considere um sinal discreto, real e finito x, representado por um vetor coluna $N \times 1$. Os sinais no espaço R^N podem ser representados a partir de projeções nas bases ortonormais ψ_i, sendo assim, x pode ser representado da seguinte maneira:

$$x = \sum_{i=1}^{N} \beta_i \psi_i = \Psi\beta. \tag{2.10}$$

O vetor β será K-esparso caso possua N−K valores próximos de zero ou iguais a zero. A teoria sobre *Compressed Sensing* prova que é possível reconstruir o sinal a partir de um vetor y que é a projeção de x em bases Φ, havendo M bases para a projeção, assim

$$y = \Phi x = \Phi\Psi\beta = \Theta\beta. \tag{2.11}$$

A matriz Φ será chamada de *Sensing Matrix*, e por não ser uma matriz reversível, o sinal $x = \Phi^{-1}\Theta\beta$ será reconstruído pela resolução do seguinte problema de otimização convexa:

$$\arg_\beta \min \|\beta\|_1 \quad \text{sujeito a} \quad \Theta\beta = y, \tag{2.12}$$

em que $||\beta||_1 = \sum_{i=1}^{N} |\beta_i|$ é a norma ℓ_1 de β.

Para utilizar essa otimização para reconstrução de sinais, duas condições precisam ser satisfeitas:

1. $M > K \log(\frac{N}{K})$;
2. Satisfazer a condição RIP (do inglês *Restricted Isometry Property*): $(1-\delta_k)\|\beta\|_2 \leq \|\Phi\Psi\beta\|_2 \leq (1+\delta_k)\|\beta\|_2$, sendo a constante δ_k não tão próxima de 1.

Neste trabalho foram usadas duas técnicas baseadas em matrizes aleatórias, uma matriz uniforme e uma matriz não uniforme, uma vez que matrizes com elementos independentes e identicamente distribuídos (i.i.d) respeitam a condição RIP. O trabalho que descreve essas técnicas é a referência [8].

2.6.1 Matriz Uniforme

A matriz Φ é dada por elementos {0,1}, por isso é denominada matriz binária, sendo que esses elementos seguem uma distribuição de probabilidade de Bernoulli i.i.d. Por exemplo, uma matriz com probabilidade equivalente a 0,5:

$$\Phi = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}.$$

2.6.2 Matriz Não Uniforme

Nesta técnica é necessário localizar uma região de interesse, que será demarcada por uma janela L. Por exemplo, a região de interesse para um sinal eletrocardiográfico pode ser o complexo QRS. Após localizar a região de interesse será elaborada a matriz Φ, sendo essa composta por três partes, ilustrado na Figura 2.7, lembrando que essa será de tamanho $M \times N$.

A parte **0** da matriz é composta somente por elementos nulos e terá tamanho $M_1 \times (N - L)$, a parte Φ_1 é uma matriz binária uniforme, com probabilidade p_1, de tamanho $M_1 \times L$ e a parte

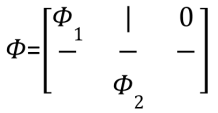

Figura 2.7: Matriz Φ.

Φ_2 é uma matriz binária uniforme, com probabilidade p_2, de tamanho $M_2 \times N$. Para adaptar a matriz para os sinais eletrocardiográficos que irão realizar a multiplicação matricial só é necessário rotacionar os elementos superiores a fim de coincidir a parte Φ_1 com a região de interesse.

Neste trabalho, os valores utilizados foram L igual a 50, p_1 igual a 0,3, p_2 igual a 0,4, M_1 igual a 36 e M_2 igual a 36, os quais foram retirados do artigo [8].

2.7 Compressão de Sinal Eletrocardiográfico via DCT

O artigo [9] foi estudado a fim de verificar uma maneira, baseada no cálculo da DCT, para realizar compressões em sinais eletrocardiográficos. Essa implementação foi utilizada neste trabalho a fim de estudar os gastos de energia que existiriam para realizar esse processamento. Sendo assim, nesta seção é apresentada essa estratégia de compressão.

Dado um conjunto de vetores $\{X\}$, pode-se representar um vetor X por

$$X = \sum_{i=1}^{N} y_i.\psi_i = \Psi.Y \tag{2.13}$$

e por sua vez

$$Y = \sum_{i=1}^{N} x_i.a_i = A.X. \tag{2.14}$$

No caso, tanto Ψ e A são matrizes $(N \times N)$ e X^T e Y^T são vetores $(1 \times N)$. No artigo, são ilustradas as características da Transformada Karhunen–Loève, e, por fim, comenta-se sobre o critério da variância onde são retirados alguns componentes da Transformada Karhunen–Loève a fim de representar o sinal utilizando a menor quantidade de componentes e com o menor erro possíveis.

Entretanto, essa transformada exige um gasto computacional muito grande, pois as matrizes Ψ e A dependem dos autovetores da matriz de covariância do conjunto de vetores $\{X\}$, assim, para cada conjunto de vetores $\{X\}$ há uma matriz de covariância distinta e, consequentemente, autovetores e matrizes Ψ e A distintos. Portanto, as matrizes Ψ e A devem ser ajustadas em tempo real uma vez que dependem dos sinais que são processados. Sendo assim é proposto realizar o critério da variância para transformadas que exigem um gasto computacional menor. No caso, é utilizado a Transformada de Cosseno e Transformada Haar. Dado o conjunto de vetores $\{Y\}$ composto pelas transformadas dos vetores X pertencentes a $\{X\}$, realiza-se o cálculo da variância para os componentes, e por fim são descartados os componentes que possuem menor variância.

Por exemplo, pretende-se ter uma compressão de 2:1, e há o seguinte vetor

$\sigma_i^2 = [12,06\ \ 3,15\ \ 1,05\ \ 4,05]^T$. Neste caso, deve-se descartar o segundo e terceiro componentes da matriz $\{Y\}$, pois são os menores valores.

2.8 Conclusão

Neste capítulo foi exposta a teoria necessária para a realização deste trabalho, indicando os modelos de consumo de energia apresentados na literatura que foram utilizados nesta monografia, assim como explicitando os conceitos básicos relativos a norma IEEE 802.15.4 e sobre o *TinyOS* e *NesC*. Por fim, foi levantada a teoria necessária para as estratégias de compressão utilizadas nos experimentos.

Capítulo 3

Descrição dos Experimentos

Neste capítulo, ocorre a apresentação e explicação dos experimentos realizados, sendo estes estimativas do consumo de energia ou experimentos práticos.

3.1 Introdução

Os experimentos aqui desenvolvidos têm como objetivo verificar os modelos de consumo de energia para redes de sensores, assim como propor modificações para tornar o modelo mais realista para o caso do gasto de energia relacionado à transmissão dos dados. Além do mais, tem-se como objetivo verificar a capacidade de processamento em *motes* disponíveis no mercado a fim de possibilitar uma redução no consumo de energia de transmissão, possibilitando assim, uma redução no consumo de energia total.

Sendo assim, este capítulo é dividido em duas partes: a primeira realiza uma explicação do experimento utilizado para analisar o gasto de energia a partir dos modelos existentes, e a segunda é uma implementação de uma aplicação onde é comparada a situação em que não há processamento da informação para reduzir os dados a serem transmitidos e a situação em que há processamento da informação para reduzir os dados a serem transmitidos.

Na primeira parte, foi imaginado um cenário de uma rede corpórea (BAN do inglês *Body Area Network*), em que sinais eletrocardiográficos são adquiridos, processados e transmitidos. O cálculo da aquisição de dados foi baseado no trabalho [3] e o de processamento foi baseado no trabalho [2]. Já para a parte de transmissão, foi proposta uma equação que leva em consideração as características da Norma IEEE 802.15.4 e as características do *transceiver* de rádio do *mote MICAZ*.

Na segunda parte, foram realizadas quatro aplicações com *motes MICAZ*: aquisição de dados de temperatura, aquisição de dados de luminosidade, aquisição de dados de temperatura com posterior processamento e aquisição de dados de luminosidade com posterior processamento. O processamento utilizado foi a DCT na qual os coeficientes não nulos são transmitidos. Essa transformada foi implementada de acordo com o algoritmo rápido descrito no trabalho [7] e na Subseção 2.5.1.

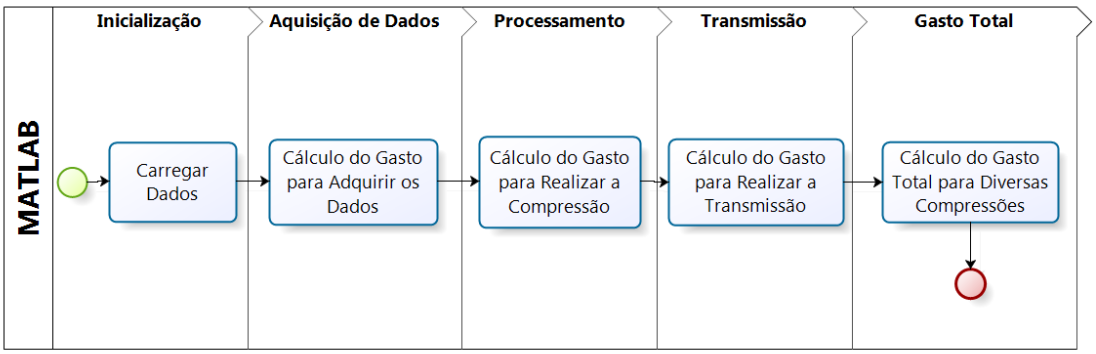

Figura 3.1: Montagem do Experimento no *Matlab*.

Para as aplicações em que houve processamento, foram realizadas transformadas que resultassem em 32 coeficientes. Este número foi escolhido pois o algoritmo proposto em [7] não faz normalizações em certas etapas do cálculo a fim de reduzir a quantidade de operações realizadas. Sendo assim, os coeficientes calculados podem extrapolar o tamanho das variáveis do microcontrolador, e apesar do *TinyOS* ter em sua documentação a possibilidade de utilização de variáveis de maior comprimento, como variáveis de 32 ou até 64 bits, o microcontrolador gera erros com essas variáveis em cálculos simples, como por exemplo, operações bit a bit.

3.2 Experimento - *Matlab*

Os experimentos do *Matlab* foram realizados a fim de verificar os gastos de energia que poderiam ocorrer numa rede BAN, observando possíveis diminuições de gastos que aconteceriam através da realização de compressões em sinais eletrocardiográficos para diminuir o montante de dados a ser transmitido.

O sinal utilizado foi o de número 103 obtido no banco de dados *MIT-BIH Arrhytmia Database* do *PhysioBank* [10]. Esse sinal foi processado, através do programa *EcgConverter*[1], a fim de ser utilizado no *Matlab*.

Conforme ilustrado na Figura 3.1, três aspectos principais foram considerados para o gasto total: gasto pela aquisição de dados, gasto pelo processamento do sinal e gasto pela transmissão do sinal.

A Equação (2.4) foi utilizada para realizar o cálculo do gasto de energia para aquisição de dados. O valor de tensão e corrente considerados foram obtidos do manual [11] e o fator de utilização foi considerado igual a um, uma vez que o equipamento é analógico e não é um elemento passivo, sempre drenando corrente. Já a quantidade de bits foi desconsiderada, uma vez que o equipamento é analógico.

A Equação (2.3) foi utilizada para realizar o cálculo do gasto de processamento. Foram consideradas as seguintes estratégias:

[1]Esse programa foi obtido em `http://www.pgea.unb.br/~joaoluiz/`

- Transmissão do sinal sem compressão;
- Transmissão do vetor obtido após realização de estratégia com *Compressed Sensing* uniforme;
- Transmissão do vetor obtido após realização de estratégia com *Compressed Sensing* não uniforme;
- Transmissão dos coeficientes, obtidos a partir da DCT, com maior variância.

Na primeira situação, por não haver método de compressão, foi considerado um gasto por processamento nulo. Para a segunda e terceira situações, tem-se que o gasto é equivalente a de um filtro FIR (do inglês *Finite Impulse Response*), uma vez que as operações das estratégias baseadas em *Compressed Sensing* são multiplicações matricias. Assim sendo, a quantidade de somas e multiplicações é equivalente a de um filtro FIR. Entretanto há um aumento da utilização da memória para armazenar todos os elementos presentes nas matrizes.

Por fim, na última situação, foi considerado o gasto de energia para a realização de uma DCT e o gasto para calcular as variâncias dos componentes através de filtros FIR a fim de verificar quais componentes são transmitidos. Tendo em mente essas considerações, os parâmetros para a o cálculo de consumo de energia foram obtidos nos artigos [2] e [3], e nos manuais [12], [13] e [1].

Já para o gasto de transmissão, foi proposta a seguinte equação, baseando-se na norma IEEE 802.15.4 e no *transceiver* de rádio [14]:

$$E_{Tx} = \gamma.V_{supA}.I_{TxA} + (1-\gamma).V_{supI}.I_{TxI} \tag{3.1}$$

em que

$$\gamma = \frac{(\frac{bits}{97.8}).(6+25).(8) + bits}{250000}. \tag{3.2}$$

Nessas equações, E_{Tx} representa o gasto de energia para transmissão em joules, V_{supA} representa a tensão em volts fornecida quando o *transceiver* de rádio está ativo e V_{supI} representa a tensão em volts fornecida quando o *transceiver* de rádio está inativo. Por sua vez, I_{TxA} representa a corrente drenada em ampères quando o *transceiver* de rádio está ativo e I_{TxI} representa a corrente drenada em ampères quando o *transceiver* de rádio está inativo. Por fim, γ representa o fator de utilização de rádio.

A fórmula do fator de utilização γ foi proposta considerando a norma IEEE 802.15.4 e características de *motes MICAZ*. O primeiro termo da soma no numerador representa a quantidade de bits de cabeçalho da camada *MAC* e da camada *PHY* a serem transmitidos e o segundo termo representa a quantidade de bits de dados que serão transmitidos. Já o denominador representa a taxa de transmissão da norma que é igual a 250kbps.

Por fim, os valores utilizados são resumidos na Tabela 3.1 e na Tabela 3.2.

Tabela 3.1: Valores para *Matlab*.

	Aquisição	**Processamento**	**Transmissão**
V_{sup}	4,8 V	2,7 V	1,8 V
I	1,4 mA	-	17,4 mA / 426 μA
γ	1	-	variável
I_o	-	1,196 mA	-
n	-	21,26	-
V_t	-	0,2 V	-
f	-	191,42 MHz	-

Tabela 3.2: Valores - Processamento.

	FIR	**DCT**
C_{tot}	0,67 mF	0,05 mF
N	$0{,}97.10^6$	$0{,}08.10^6$

3.3 Experimentos Práticos - *TinyOS*

Para a implementação dos experimentos práticos, foram utilizados *motes MICAZ* e a placa *MDA100CB* [15], ambos da *Crossbow*. Para a programação dos *motes* foi utilizado o *TinyOS*. Foram realizados quatro experimentos:

- Aquisição de dados de temperatura;
- Aquisição de dados de temperatura utilizando a transformada rápida de cosseno para reduzir o montante de dados a ser transmitido;
- Aquisição de dados de luminosidade através de fotoresistor;
- Aquisição de dados de luminosidade através de fotoresistor utilizando a transformada rápida de cosseno para reduzir o montante de dados a ser transmitido.

A partir desses experimentos, foi realizado o levantamento dos seguintes dados:

- Levantamento da corrente elétrica;
- Levantamento da energia dissipada;
- Levantamento da quantidade de bits transmitidos nos casos em que há utilização da transformada rápida e no caso em que não se utiliza a transformada rápida;
- Comparação dos sinais recebidos ao utilizar a transformada rápida e não utilizando a transformada rápida.

A fim de adquirir os dados de corrente de maneira automatizada, foi utilizado o equipamento *Agilent 34410A* [16]. Esse é um multímetro digital que possui as seguintes características principais:

Tabela 3.3: Configuração do Multímetro *Agilent*.

Atributo	**Valor**
Leitura	DCI
Escala	100 mA
Auto Zero	Desligado
Medida de Pico	Desligado
Trigger	Externo
Contador de *Trigger*	Um
Curva de *Trigger*	Negativa
Atraso de *Trigger*	Auto
Intervalo entre amostras	0,659631 ms
Quantidade de amostras	50000

- Possibilita 10000 leituras/segundo com precisão de 5 1/2 dígitos;
- Possibilita 1000 leituras/segundo com precisão de 6 1/2 dígitos;
- Padrões LAN (do inglês *Local Area Network*), USB (do inglês *Universal Serial Bus*) e GPIB (do inglês *General Purpose Interface Bus*);
- Armazenamento de até 50000 leituras em memória não volátil;
- Medições: DCV, ACV, DCI, ACI, Resistência, Frequência, Período, entre outras.

Outro fator que colaborou na escolha desse equipamento é a existência de um servidor web embutido nele, assim ao acessá-lo via LAN pode-se realizar todas as configurações necessárias para a realização das leituras, assim como transmitir os dados para o computador pessoal através dessa conexão.

Já para adquirir medidas de tensão das pilhas, foi utilizado um equipamento mais simples, uma vez que a tensão da bateria não irá variar tanto quanto a corrente elétrica. O equipamento utilizado foi o multímetro digital *HM-1000* da Hikari [17], possuindo as seguintes características:

- Possibilita 2000 contagens com precisão de 3 1/2 dígitos;
- Medições: DCV, ACV, DCI e Resistência.

A montagem do experimento está ilustrada na Figura 3.2.

Com a intenção de adquirir 10 ciclos de transmissão para o levantamento dos dados, a configuração do multímetro digital da *Agilent* está na Tabela 3.3.

3.3.1 Aquisição de Dados de Temperatura

Para facilitar a compreensão desse experimento, deve-se dividí-lo no receptor e no transmissor.

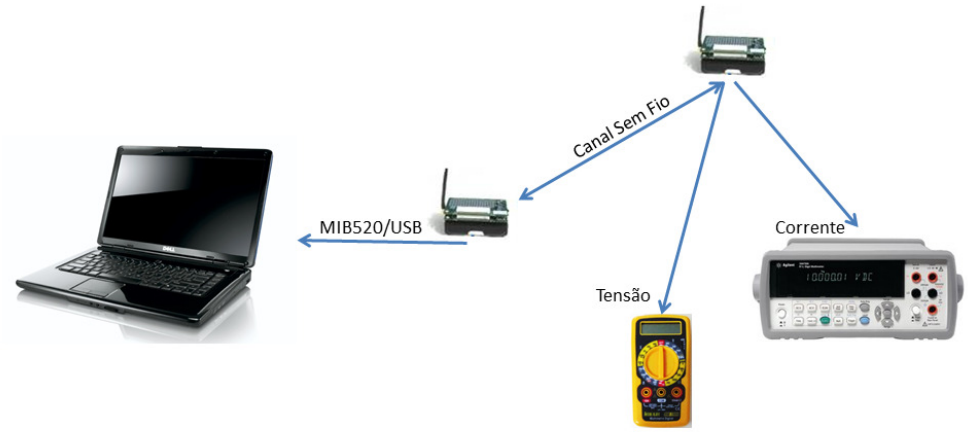

Figura 3.2: Montagem do Experimento Prático.

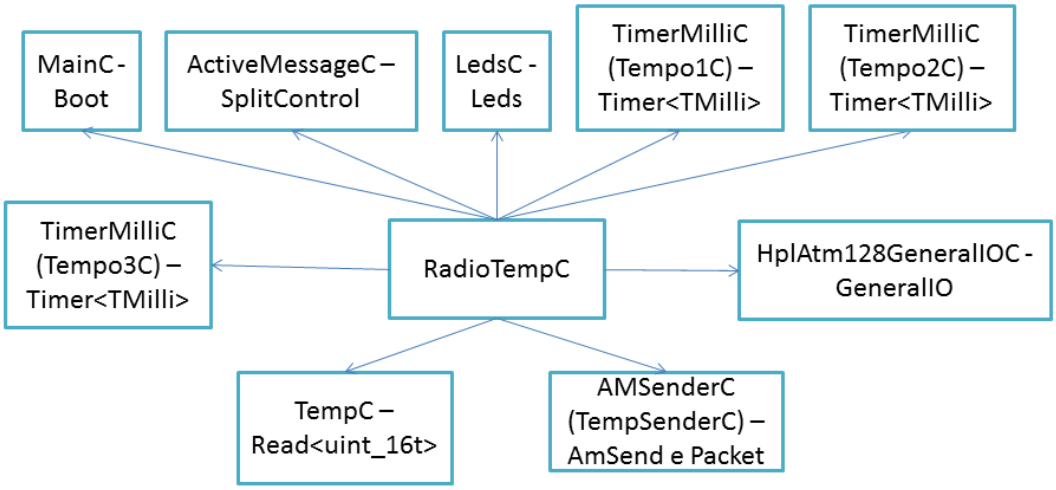

Figura 3.3: Nó Transmissor para Aquisição de Temperatura.

3.3.1.1 Nó Transmissor

Para o transmissor, leva-se em consideração a Figura 3.3.

Conforme comentado na introdução teórica, *TinyOS* trabalha com interfaces e componentes. Para a criação dessa aplicação, foram utilizadas interfaces de componentes já existentes. As interfaces são responsáveis pela notificação de eventos concluídos ou chamadas para execução de eventos, possibilitando assim, uma visão em alto nível do sistema. Como exemplo, pode-se citar: realização

de leitura, inicialização ou interrupção de rádio, inicialização dos componentes do sistema, entre outros. Foram utilizadas as seguintes interfaces:

- *Boot* - responsável pela notificação de que os componentes foram iniciados e o sistema está apto para uso;
- *SplitControl* - responsável pelo controle do rádio, ligando e desligando o *transceiver* de rádio possibilitando assim a economia de energia;
- *Leds* - responsável pela indicação de quando o rádio está ativo;
- *Timer<TMilli>* - há três instâncias para essa interface, uma instância é responsável pelo período de amostragem do sinal de temperatura, e as outras duas são responsáveis pela duração de um pulso que é utilizado para o *Trigger* do multímetro;
- *Read* - responsável pela leitura do sinal de temperatura;
- *AMSend* - responsável pelo envio da mensagem via rádio;
- *Packet* - responsável pelo preparo do pacote a ser enviado via rádio;
- *GeneralIO* - responsável pelo pino que gera um pulso que é utilizado para o *Trigger* do multímetro.

Os componentes por sua vez realizam a ponte entre essa visão alto nível para baixo nível, ou seja, são responsáveis pelo controle do *Hardware*. Sendo assim, ao utilizar componentes já existentes, facilita-se a implementação da aplicação. Foram utilizados os seguintes componentes [6];

- *MainC*;
- *ActiveMessageC*;
- *LedsC*;
- *TimerMilliC*;
- *TempC*;
- *AMSenderC*;
- *HplAtm128GeneralIOC*.

Na Figura 3.4, há uma ilustração do que ocorre no *mote* transmissor.

Inicialmente, após ligar o *mote*, há um evento que notificará a inicialização dos componentes. A partir desse evento é preparado um pulso que é utilizado como *Trigger* para o multímetro, para inicializar a coleta da corrente elétrica.

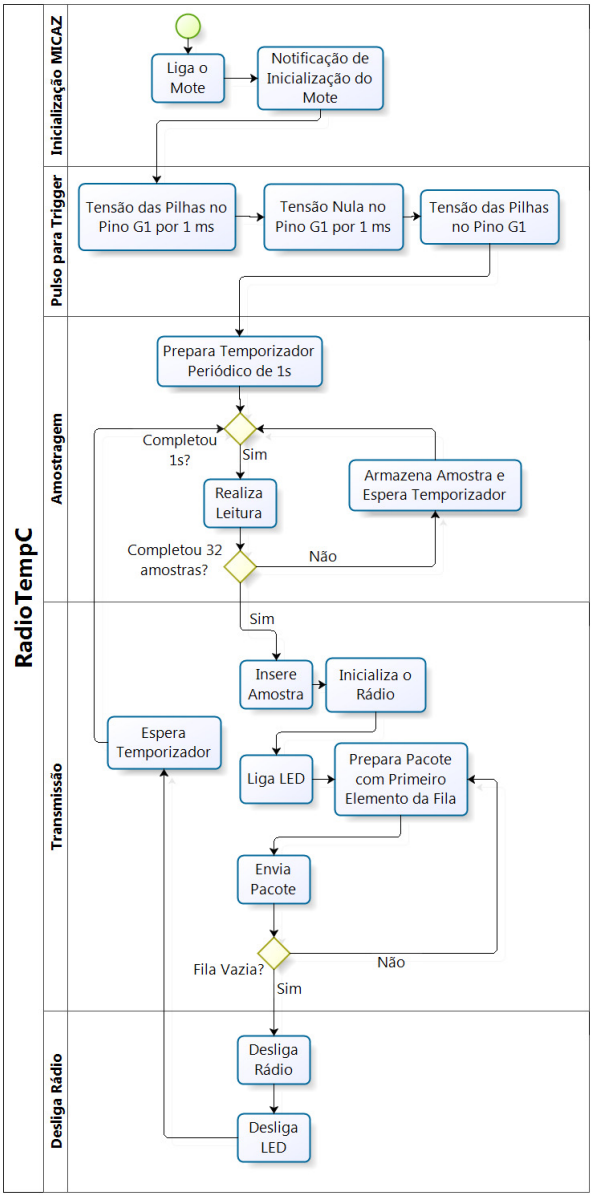

Figura 3.4: Fluxograma da Aplicação para Aquisição de Temperatura.

Posteriormente, é configurado um temporizador periódico de 1 segundo. Quando esse tempo for finalizado, uma amostra de temperatura é lida. Caso essa amostra não seja a trigésima segunda, deve-se esperar a próxima leitura. Caso a amostra seja a trigésima segunda, deve-se ligar o rádio e transmitir amostra por amostra em pacotes. Ao transmitir todas as amostras, deve-se desligar o rádio e esperar a coleta de novas 32 amostras.

A amostragem é realizada pelo termistor YSI 44006 [18] presente na placa *MDA100CB* da *Crossbow*. Esse termistor varia sua resistência conforme a temperatura ambiente. O valor a ser convertido pelo conversor analógico-digital é dado pelo divisor de tensão ilustrado na Figura 3.5.

Para obter o valor da temperatura em kelvin, a partir da leitura do conversor analógico-digital, e reconstruir o sinal após a recepeção dos pacotes é necessário utilizar a seguinte fórmula, obtida

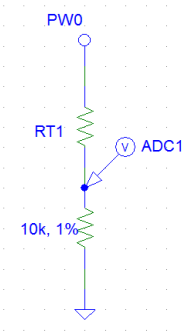

Figura 3.5: Divisor de Tensão do Termistor.

em [15]:

$$\frac{1}{T(K)} = 0,001010024 + 0,000242127.\ln(10000.(1023 - Valor)/Valor) + 0,000000146.[\ln(10000.(1023 - Valor)/Valor)]^3. \quad (3.3)$$

Um ciclo de transmissão é composto por essa coleta de 32 amostras seguida da transmissão desses valores. O ciclo foi proposto dessa maneira a fim de tornar a comparação entre o método sem a transformada de cosseno com o método da transformada de cosseno mais justa, evitando assim ligar e desligar o rádio toda vez que uma amostra fosse coletada, gerando um gasto de energia desnecessário.

Por fim, o pacote transmitido é composto pelos seguintes campos, conforme representado na Figura 3.6:

- Um byte indicando que é um pacote do tipo *ActiveMessage*;
- Dois bytes indicando o nó de destino;
- Dois bytes indicando o nó de origem;
- Um byte indicando o tamanho do *payload*;
- Um byte indicando o ID do grupo de origem;
- Um byte identificando qual tipo de pacote é dentre os tipos de *ActiveMessage*;
- Dois bytes para a amostra que é transmitida.

3.3.1.2 Nó Receptor

Para o receptor, leva-se em consideração a Figura 3.7.

A aplicação *BaseStationC* é disponibilizada pelos desenvolvedores do *TinyOS*, sendo essa responsável pela interface entre a rede de sensores com o computador pessoal. O nó que possuir

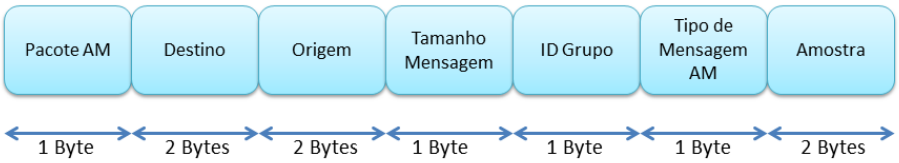

Figura 3.6: Pacote AM da Aplicação para Aquisição de Temperatura.

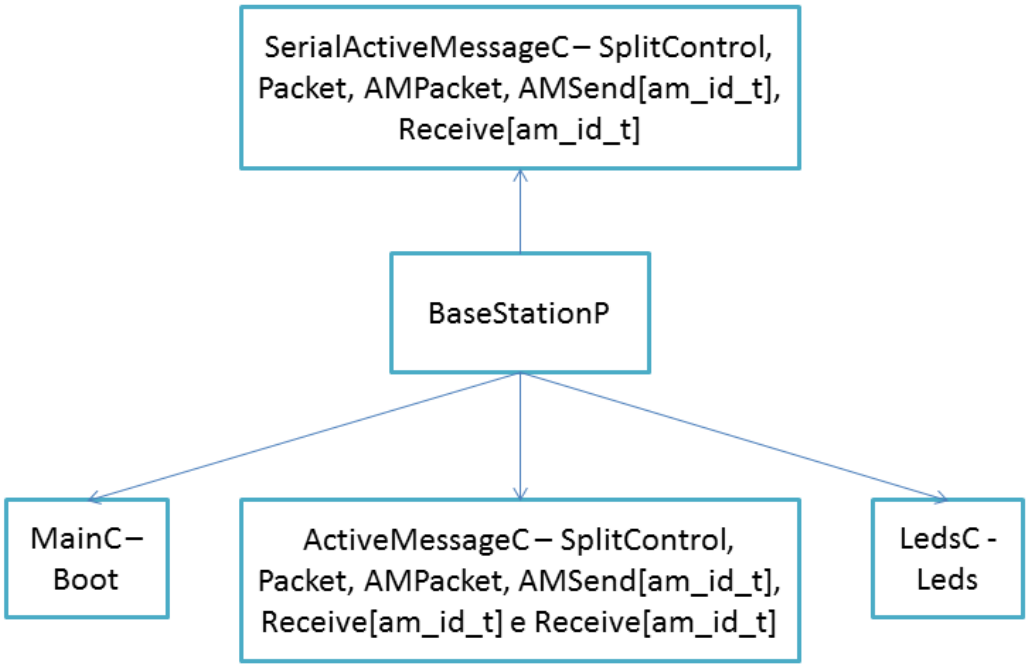

Figura 3.7: Nó Receptor para Aquisição de Temperatura.

instalado essa aplicação poderá enviar mensagens do computador pessoal para a rede de sensores para, por exemplo, atualizar parâmetros de configuração dos *motes* presentes na rede de sensores. O nó também poderá receber as mensagens dos *motes* e passá-las para aplicações no computador pessoal. Esta ligação do nó com o computador é realizada pela placa *MIB520* [19], através de uma conexão USB. A placa em questão está ilustrada na Figura 3.8.

Conforme pode ser visto na Figura 3.7, o nó receptor usará as seguintes interfaces:

- *Boot*;
- *Leds*;
- *AMPacket*;
- *SplitControl*;
- *Packet*;

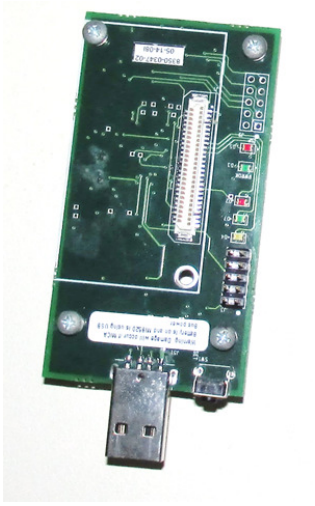

Figura 3.8: Placa MIB 520.

- *Receive*;
- *AMSend*;
- *AMSend*.

As características das interfaces serão semelhantes ao nó transmissor. Todavia, agora também há a responsabilidade da comunicação serial, havendo interfaces para recebimento e envio de pacotes dessa.

Os componentes utilizados para essa aplicação são:

- *MainC*;
- *ActiveMessageC*;
- *SerialActiveMessageC*;
- *LedsC*.

Por fim, utiliza-se o programa *Listen*, também disponibilizado pelos desenvolvedores do *TinyOS*, para exibir os pacotes que estão chegando via serial no terminal do computador. Os pacotes são salvos em um arquivo de texto para análise posterior.

3.3.2 Aquisição de Dados de Luminosidade

No segundo experimento, a parte do receptor é exatamente igual ao do primeiro experimento. Já o transmissor terá um componente diferente conforme pode ser visto na Figura 3.9.

No caso, há a substituição do componente *TempC* pelo componente *PhotoC*. Assim, é possível realizar a leitura da queda de tensão no fotoresistor presente na placa *MDA100CB*. O fotoresistor

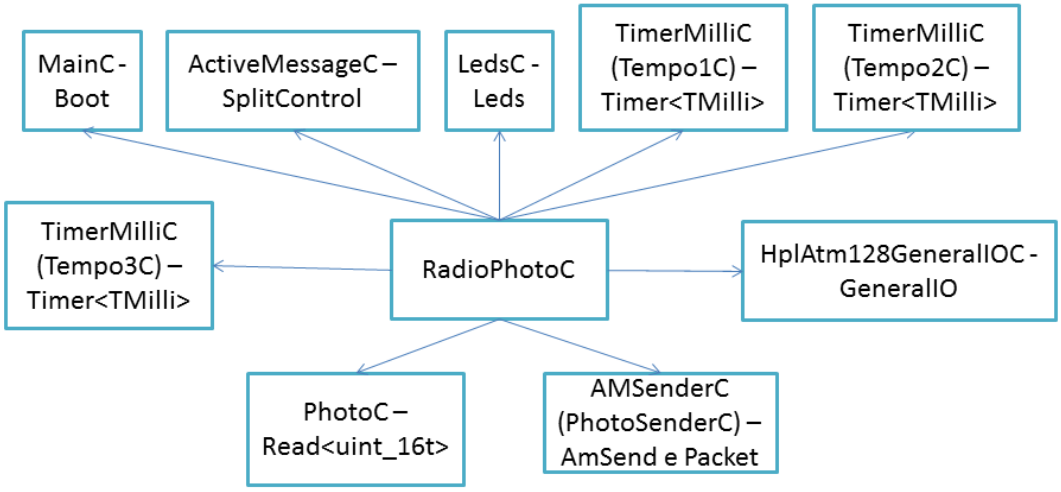

Figura 3.9: Nó Transmissor para Aquisição de Luminosidade.

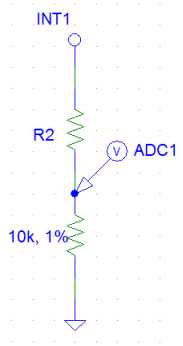

Figura 3.10: Divisor de Tensão do Fotoresistor.

varia sua resistência de acordo com a luz presente no ambiente, quanto maior a intensidade de luz no ambiente, menor é a resistência do fotoresistor. O valor a ser convertido pelo conversor analógico-digital é dado pelo divisor de tensão ilustrado na Figura 3.10.

Como não há uma fórmula específica para associar o decaimento de tensão com a intensidade da luz ambiente, o sinal reconstruído na análise de dados é um sinal do valor de tensão entre os dois elementos da Figura 3.10. Para obter o valor de tensão a partir do valor dado pelo conversor analógico-digital, utiliza-se a seguinte fórmula

$$V_{foto} = \frac{Valor.V_{ref}}{1024}. \tag{3.4}$$

Sendo V_{foto} a tensão entre os elementos, $Valor$ é o número dado pelo conversor e V_{ref} é o valor de tensão das pilhas.

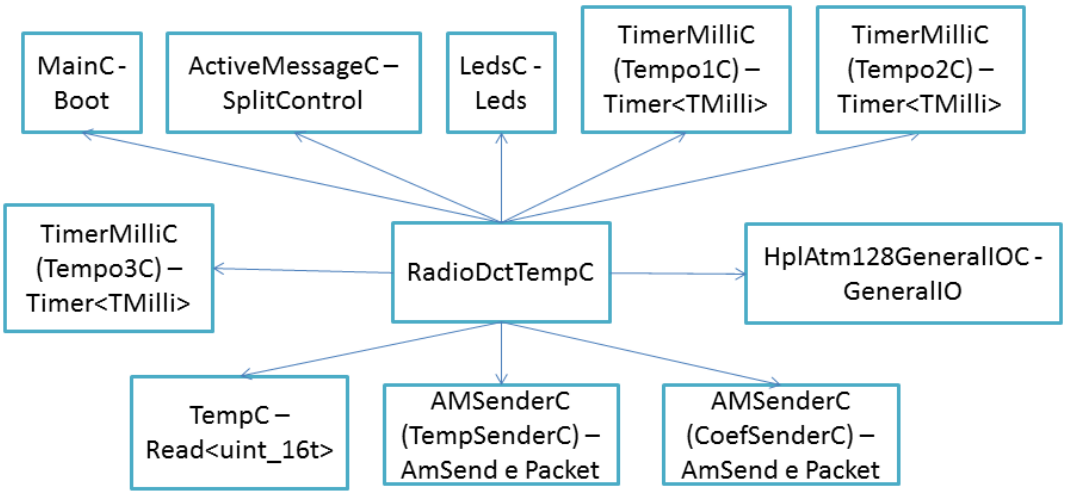

Figura 3.11: Nó Transmissor para Aquisição de Temperatura com DCT.

3.3.3 Aquisição de Dados de Temperatura com DCT

Para o terceiro experimento, o receptor permanece equivalente aos anteriores. E o transmissor é esquematizado de acordo com a Figura 3.11.

Comparado com o experimento em que não é executado a DCT, há a existência de um componente e de duas interfaces a mais. Isso ocorre pois é necessário utilizar dois tipos de pacote, um para transmitir variáveis que indicam quais coeficientes são nulos e quais são não nulos e outro para transmitir os valores dos coeficientes não nulos. Na Figura 3.12, há um fluxograma para explicar o que ocorre nessa aplicação.

O cálculo da DCT é realizado logo após efetuar as 32 amostragens, foi implementado o algoritmo rápido proposto em [7], já explicado no Capítulo 2. Após o cálculo dos coeficientes da DCT, realiza-se a conversão desses que são do tipo *float* para valores do tipo *int*. Caso o valor *int* seja equivalente a zero, o coeficiente não é inserido na fila dos valores a serem transmitidos. E caso o valor seja diferente de zero, o coeficiente é inserido na fila de transmissão.

Além disso, para posssibilitar o cálculo da transformada inversa na recepção, são utilizadas duas variáveis de 16 bits. Basicamente, caso o coeficiente não seja nulo, configuramos o bit da variável de equivalente posição igual a 1. A primeira variável é responsável pelos coeficientes de 0 a 15 e a segunda variável é responsável pelos coeficientes de 16 a 31.

Na Figura 3.13 está ilustrado o primeiro tipo de pacote, e na Figura 3.14 está ilustrado o segundo tipo de pacote.

O cabeçalho dos pacotes é idêntico ao do pacote da aplicação em que não ocorre a DCT. No primeiro, há três campos no *payload* do pacote:

- *Variável de Coeficientes 1* - indica se os coeficientes de 0 a 15 são nulos ou não;
- *Variável de Coeficientes 2* - indica se os coeficientes de 16 a 31 são nulos ou não;

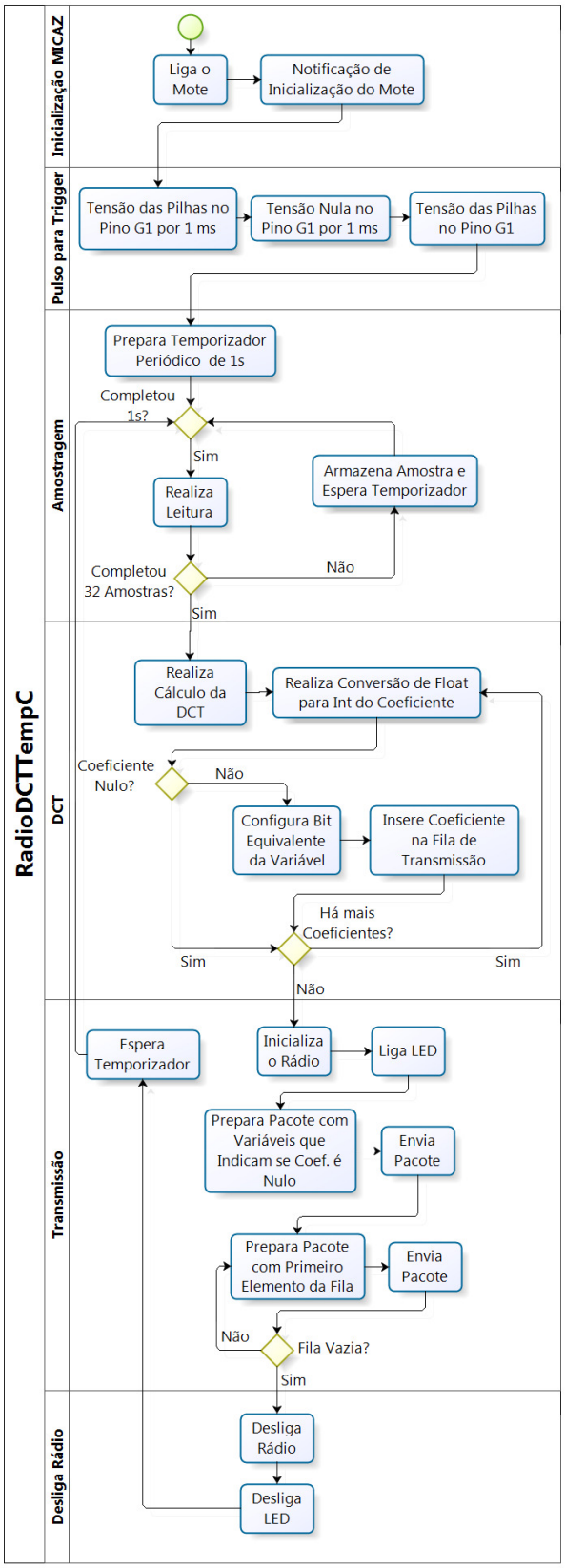

Figura 3.12: Fluxograma da Aplicação para Aquisição de Temperatura com DCT.

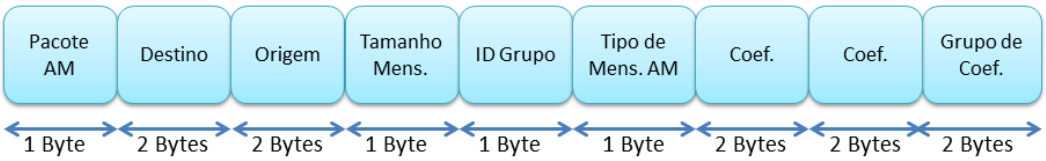

Figura 3.13: Pacote AM da Aplicação para Aquisição de Temperatura com DCT - Tipo 1.

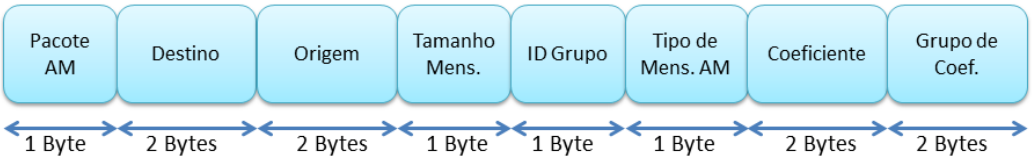

Figura 3.14: Pacote AM da Aplicação para Aquisição de Temperatura com DCT - Tipo 2.

- *Grupo de Coeficientes* - indica em que ciclo de transmissão está esse coeficiente.

No segundo, há dois campos no *payload* do pacote:

- *Valor do Coeficiente* - que indicará o valor do coeficiente;
- *Grupo de Coeficientes* - indica em que ciclo de transmissão está esse coeficiente.

3.3.4 Aquisição de Dados de Luminosidade com DCT

Para o quarto experimento, o receptor permanece equivalente aos anteriores. E o transmissor é esquematizado de acordo com a Figura 3.15.

A estrutura é semelhante ao do caso da aplicação de aquisição de temperatura com DCT, sendo a única mudança a troca do componente *TempC* para o componente *PhotoC*.

3.4 Conclusão

Neste capítulo foi detalhada a organização dos experimentos do *Matlab* e práticos. Na parte de estimação do consumo de energia, os parâmetros utilizados e a equação necessária para o consumo de energia na transmissão foram apresentados, assim como as considerações para o consumo relativo ao processamento e aquisição de dados foram explicadas. Na parte dos experimentos práticos, as aplicações implementadas foram elucidadas, sempre realizando a associação aos componentes e interfaces do *TinyOS*, além disso foi explicitada a configuração utilizada no equipamento responsável pela captura dos valores de corrente, uma vez que essa foi realizada de maneira automatizada.

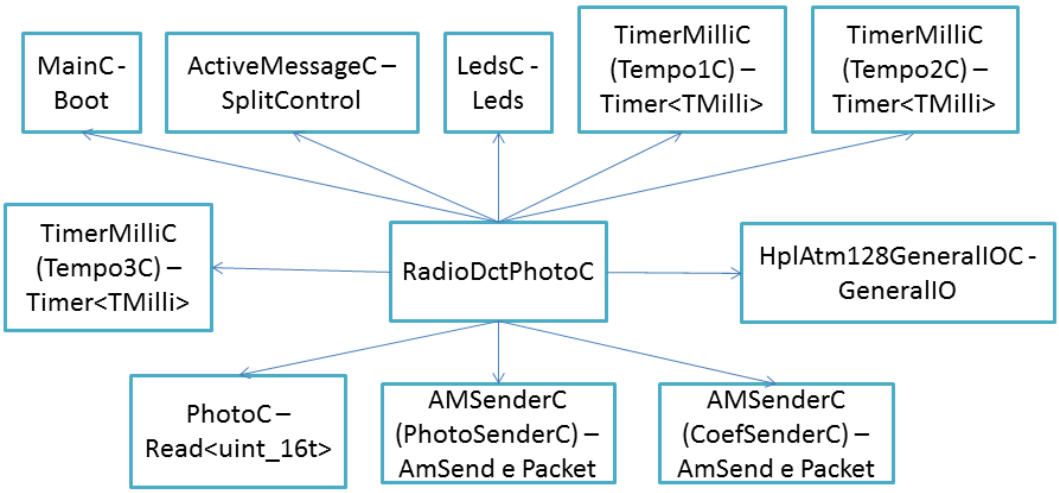

Figura 3.15: Nó Transmissor para Aquisição de Luminosidade com DCT.

Capítulo 4

Resultados Experimentais

Este capítulo apresenta os resultados obtidos durante a execução do trabalho e seus pseudo-códigos, ressaltando os pontos mais importantes.

4.1 Introdução

Neste capítulo, são apresentados os resultados obtidos através de experimentos no *Matlab* e práticos. Na parte do *Matlab*, um sinal eletrocardiográfico foi utilizado a fim de estimar o gasto de energia total para a situação em que não fosse realizado compressão e situações em que fossem efetuadas estratégias de compressão: *Compressed Sensing* e DCT com critério da variância. A partir desse gasto geral, foi especificado o percentual do gasto para cada componente: aquisição de dados, processamento e transmissão. E a partir dessa análise, foi possível planejar qual processamento poderia ser realizado no procedimento prático.

A partir do diagnóstico inicial realizado com o experimento do Matlab, foi implementada a DCT a fim de verificar os impactos no consumo de energia que essa traria nas aplicações. Além de verificar a média do consumo de energia para os casos em que houve o processamento e para os casos em que não foi realizado o processamento, também foi possível caracterizar a corrente que passou pelo sistema, verificar a redução da quantidade de bits que é transmitida e por fim analisar a reconstrução do sinal no receptor, possibilitando ver a inserção de erro ao realizar a não transmissão de determinados coeficientes.

4.2 Resultados Numéricos - *Matlab*

Os gráficos obtidos através de estimações do consumo de energia realizados no *Matlab* são comentados nesta seção. Inicialmente, é analisada a Figura 4.1, onde se pode visualizar a estimação do gasto de energia para a aquisição, processamento e transmissão do sinal eletrocardiográfico, conforme já explicado no Capítulo 3.

Na Figura 4.1 é possível visualizar que o consumo de energia total[1] do *mote* para todas as

[1]Energia total inclui aquisição, processamento e transmissão

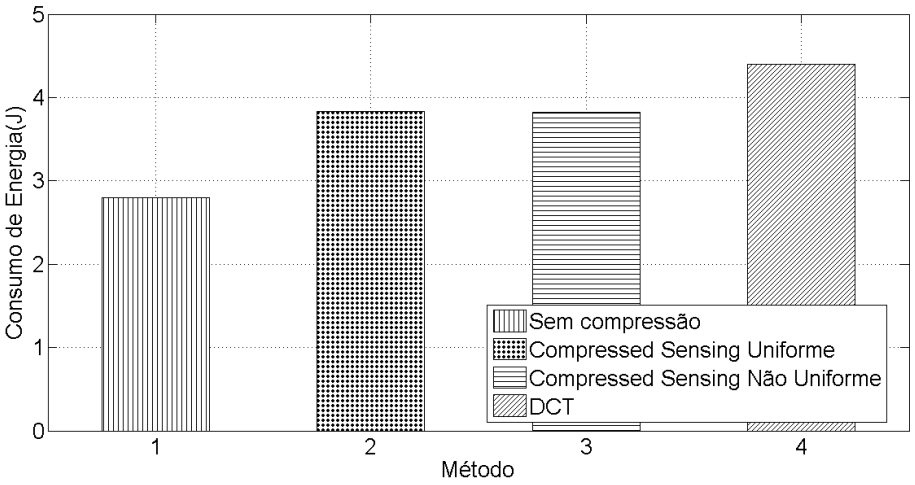

Figura 4.1: Gasto de Energia Total.

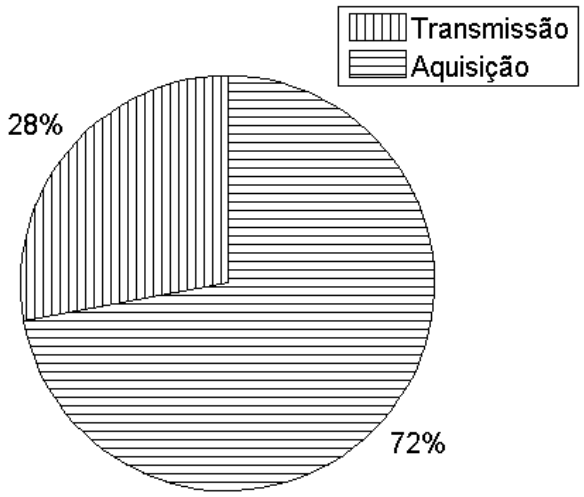

Figura 4.2: Divisão do Gasto de Energia - Sem Compressão.

situações em que há processamento foi maior do que no caso em que não há processamento do sinal. Portanto, o gasto adicional existente para realizar o processamento foi superior à economia gerada pela eliminação de dados a serem transmitidos.

Nas Figuras 4.2, 4.3, 4.4 e 4.5, são mostradas as composições desses gastos de energia para cada situação, ou seja, o percentual de participação de cada fator: aquisição, processamento e transmissão. Ao analisar esses gráficos, é possível realizar uma avaliação mais aprofundada.

O segundo gráfico a ser analisado é a Figura 4.2, onde é ilustrado a divisão do gasto de energia na situação em que não há compressão.

Como a Figura 4.2 não há compressão, o gasto de energia com o processamento foi considerado nulo, sendo os únicos fatores considerados o gasto de aquisição e o gasto de transmissão. Em um

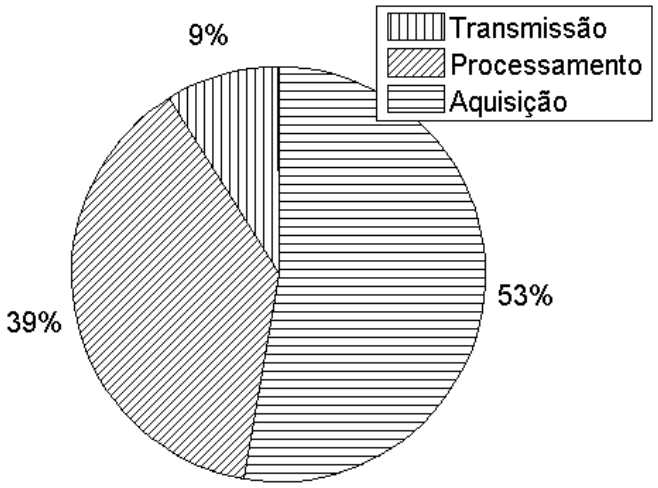

Figura 4.3: Divisão do Gasto de Energia - *Compressed Sensing* Uniforme.

primeiro momento, este gráfico pode causar certo estranhamento, pois o gasto de energia para a transmissão teve um percentual de participação no gasto de energia total muito inferior ao gasto de energia de aquisição, o que contradiz a maioria dos estudos que trabalham com os modelos de consumo de energia, entre eles [2] e [3].

Todavia, esse resultado faz sentido, pois conforme explicado no artigo [20], a maioria dos modelos de consumo de energia não consideram sensores que tenham um gasto de energia elevado, normalmente são consideradas aplicações em que se utilizam sensores simples. Entretanto, ao utilizar sensores com gasto de energia mais elevado, como o sensor eletrocardiográfico em questão, o gasto por aquisição se torna um fator preponderante para o gasto de energia total. Além do mais, conforme explicado no Capítulo 3, o gasto de transmissão foi modificado para evitar os cálculos de perda de percurso existentes e adaptado para ficar adequado as características dos *motes MICAZ* e da norma IEEE 802.15.4.

A seguir, é feita a análise das Figuras 4.3 e 4.4, uma vez que são estratégias semelhantes e são baseadas em *Compressed Sensing*.

Como já foi comentado anteriormente, percebe-se que a aquisição de dados exerceu um fator preponderante no gasto de energia. Além do mais, nas duas estratégias de processamento, realizaram-se os métodos com uma taxa de compressão equivalente a 80%, e portanto, os gastos para transmissão foram equivalentes.

Ao analisar o gasto de processamento, percebe-se que ele teve uma participação muito maior do que a transmissão para o consumo de energia total, sendo assim a ideia inicial de aumentar o consumo no processamento e diminuir o consumo na transmissão não surtiu efeito, uma vez que o aumento do processamento foi muito maior do que a diminuição do gasto de transmissão.

Por fim, a Figura 4.5 ilustra a divisão do consumo de energia quando se utiliza a estratégia

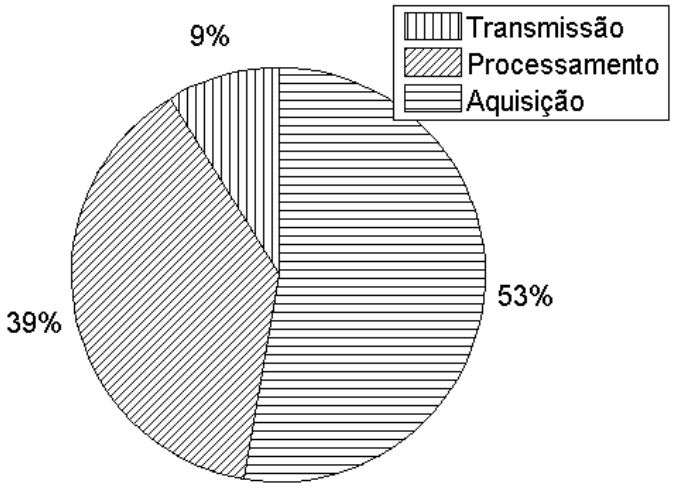

Figura 4.4: Divisão do Gasto de Energia - *Compressed Sensing* Não Uniforme.

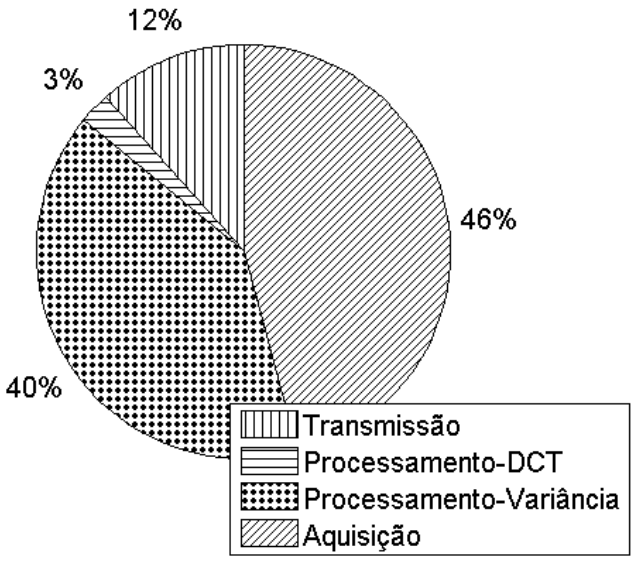

Figura 4.5: Divisão do Gasto de Energia - DCT.

com a DCT.

Ao analisar o gráfico percebe-se que o processamento foi dividido em duas partes, uma para representar os gastos existentes para realizar a DCT e a outra para realizar os cálculos com a variância necessários para descobrir quais componentes devem ser descartados, no caso, considerou-se que o gasto de energia da variância é equivalente a de um filtro FIR. Ao observar esses dois elementos nota-se que o cálculo da variância realiza um aumento significativo no gasto total do equipamento, já o cálculo da DCT representa apenas 3% do gasto total do equipamento. O baixo consumo para realizar a transformada já era esperado, uma vez que, de acordo com [2] o cálculo

Tabela 4.1: Utilização de ROM - Bytes.

	4 valores	**8 valores**	**16 valores**	**32 valores**
Temperatura	15124	15124	15124	15124
Temperatura - DCT	26284	26284	26284	26292
Luminosidade	15124	15124	15124	15124
Luminosidade - DCT	26284	26284	26284	26292

Tabela 4.2: Utilização de RAM - Bytes.

	4 valores	**8 valores**	**16 valores**	**32 valores**
Temperatura	401	409	425	457
Temperatura - DCT	1182	1198	1230	1294
Luminosidade	401	409	425	457
Luminosidade - DCT	1182	1198	1230	1294

da DCT é o que possui menor consumo de energia dentre os processamentos citados nesse artigo. Considerando esses pontos supracitados, o uso da DCT nos experimentos práticos foi realizado para verificar seus impactos.

4.3 Experimentos Práticos - *TinyOS*

Antes de analisar os gráficos obtidos com a realização dos experimentos citados no Capítulo 3, a utilização das memórias RAM e ROM é apresentada nas Tabelas 4.1 e 4.2. Deve-se ter em mente que para *motes MICAZ* há a disponibilidade de 128KB para ROM e 4KB para RAM.

Apesar da utilização das memórias nos casos em que ocorre o processamento com a DCT ser muito maior do que nos casos em que não ocorre processamento, percebe-se que a utilização da memória RAM para o caso em que é calculada a DCT com 32 valores alcança um valor de 32,35% do total de memória fornecida. Já para memória ROM, percebe-se que esse percentual é ainda menor, no caso a utilização é de aproximadamente 20,54% da memória total fornecida. Sendo assim, nota-se a possibilidade de aumentar o tamanho da transformada, mas para isso é necessário inserir as normalizações que foram retiradas do algoritmo, como já relatado no Capítulo 3.

Conforme já citado no Capítulo 3, houve a aquisição de 10 ciclos de transmissão, podendo assim, realizar um levantamento estatístico da quantidade média de energia consumida por ciclo e a quantidade média de bits trasmitidos, também realizando uma barra de erro com intervalo de confiança de 95%. Além do mais, os sinais recebidos foram reconstruídos, assim, há a possibilidade de compará-los, verificando o erro quadrático médio entre o sinal sem o processamento e o sinal com processamento da DCT.

Nas Figuras 4.6 e 4.7 é apresentada a série temporal da corrente que alimenta o *mote* para a aplicação de temperatura e para a aplicação de luminosidade, respectivamente. Ao observar essas duas figuras, é fácil notar 10 picos de corrente com valores de aproximadamente 25 mA.

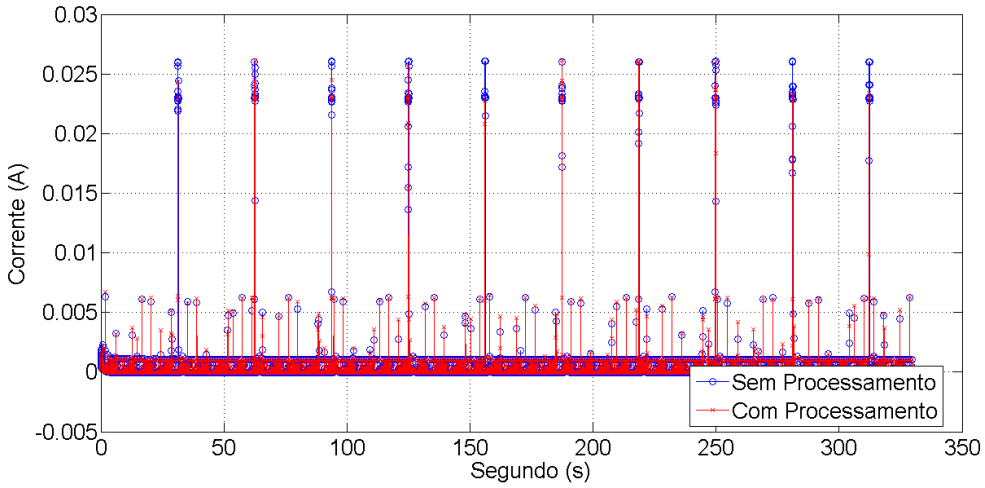

Figura 4.6: Série Temporal da Corrente para Aplicação de Temperatura.

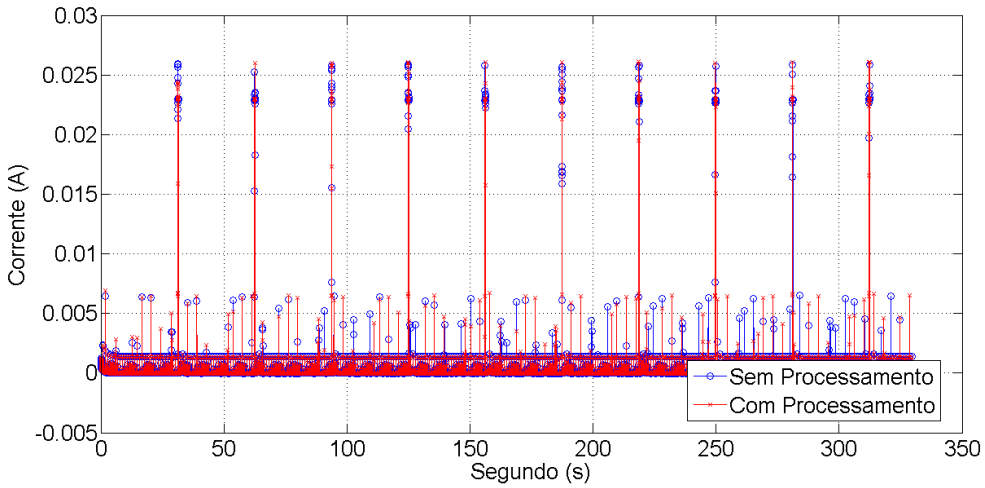

Figura 4.7: Série Temporal da Corrente para Aplicação de Luminosidade.

Esses valores de corrente são correspondentes ao momento em que se liga o rádio para realizar a transmissão dos dados, representando assim os ciclos de transmissão dos dados. Esses dados são coerentes com os valores apresentados em [14], sendo assim, percebe-se realmente que a transmissão exerce um fator preponderante no gasto de energia.

Para facilitar a análise dos picos de corrente, uma ampliação da série temporal em um dos pico de cada aplicação foi realizada conforme é visto nas Figuras 4.8 e 4.9. Ao observar essas figuras, percebe-se que a corrente de pico não é constante. Isso ocorre pois o rádio não vai estar transmitindo em todo momento. Há momentos em que está ocorrendo a formação do pacote a ser transmitido e outros momentos em que há a transmissão, gerando assim essas variações de corrente. Também é interessante verificar a diferença de largura entre os picos, no caso o pico em que não há processamento e o pico em que há processamento.

Conforme esperado reduzindo a quantidade de dados a ser transmitida, diminui-se o tempo

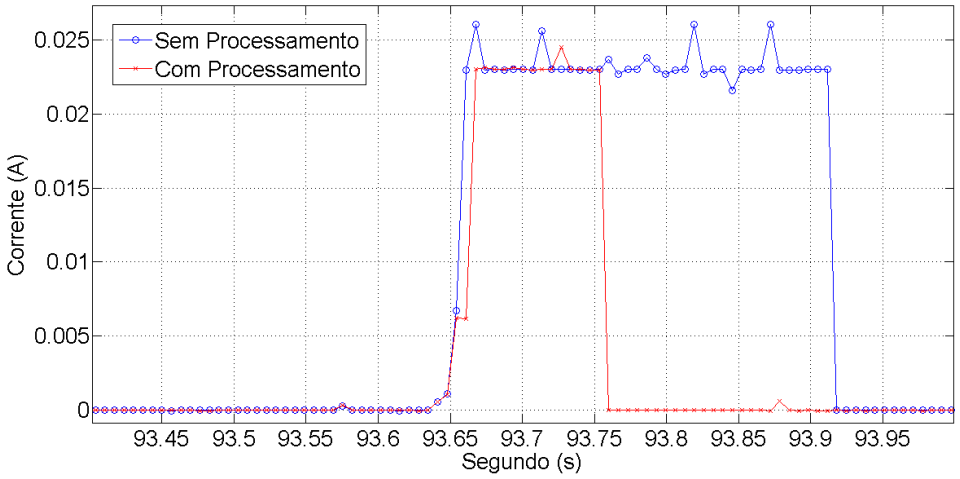

Figura 4.8: Corrente de Pico para Aplicação de Temperatura.

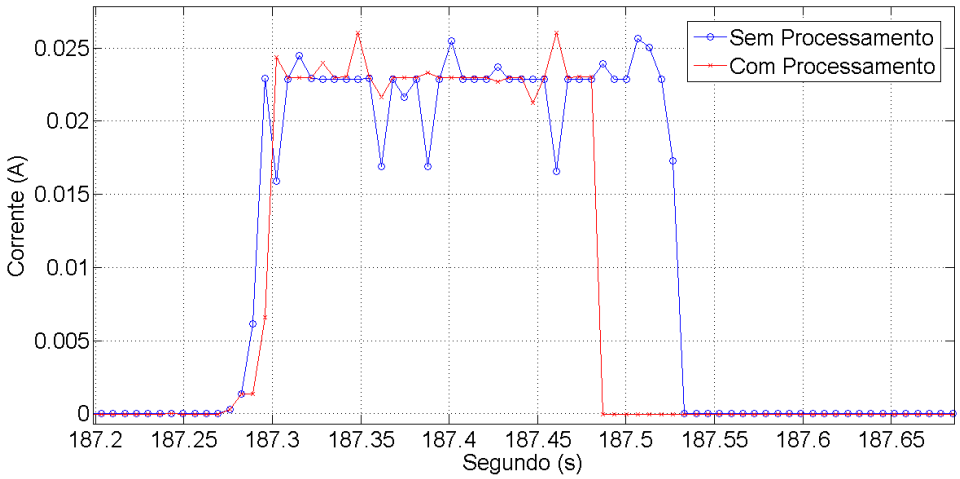

Figura 4.9: Corrente de Pico para Aplicação de Luminosidade.

que o *transceiver* de rádio permanece ligado, consequentemente, espera-se a diminuição do gasto de energia, sendo esse resultado apresentado nas Figuras 4.10 e 4.11.

Outro fator interessante a ser comentado, é que apesar do cálculo da transformada ser realizado antes de se ligar o rádio, a corrente utilizada permanece praticamente constante, assim sendo, não há um aumento significativo do consumo de energia para o cálculo da transformada conforme já observado na análise da Figura 4.5.

Os gastos de energia por ciclo de transmissão das duas aplicações estão apresentados nas Figuras 4.10 e 4.11.

Para realizar o cálculo do gasto de energia, mede-se o valor de tensão nas pilhas e a partir dos valores de corrente, realiza-se o seguinte cálculo para cada ponto de corrente obtido

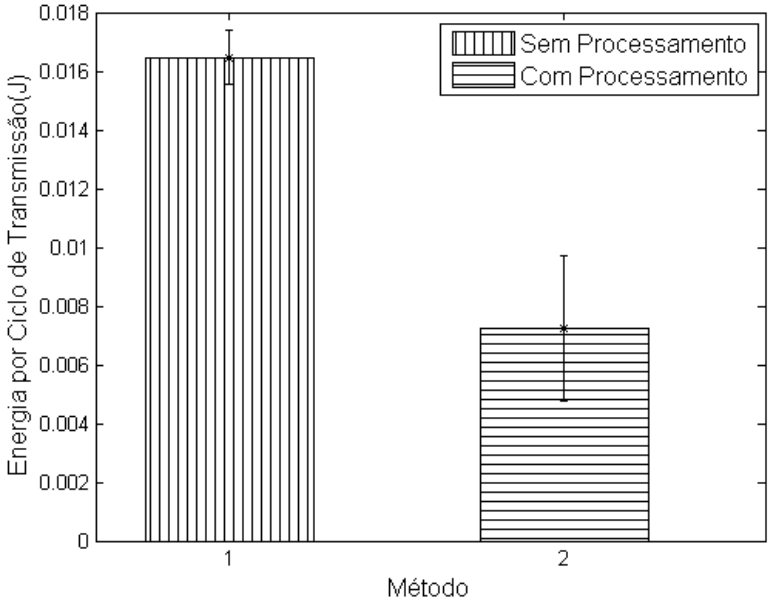

Figura 4.10: Gasto de Energia por Ciclo de Transmissão para Aplicação de Temperatura.

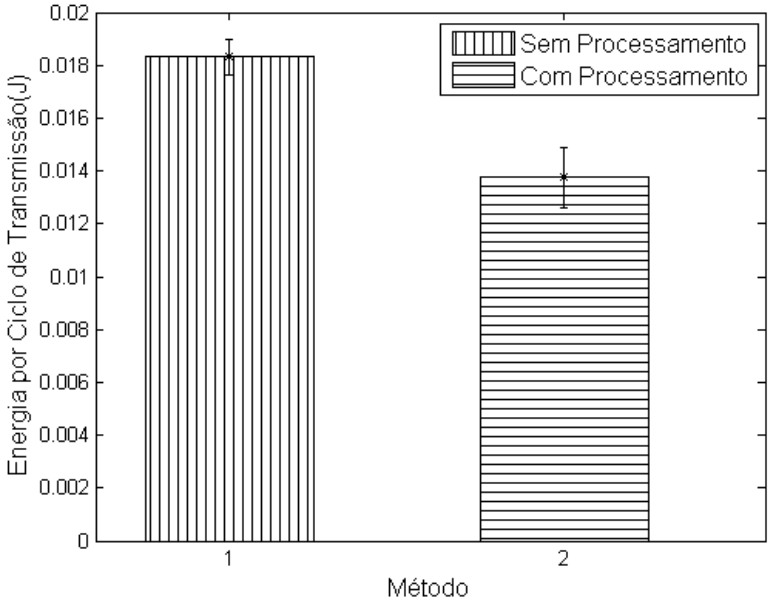

Figura 4.11: Gasto de Energia por Ciclo de Transmissão para Aplicação de Luminosidade.

$$E_{inst} = V_{sup}.I_{inst}.T_{amost}. \tag{4.1}$$

Sendo E_{inst} o valor da energia consumida em joules para aquele instante, V_{sup} o valor de tensão em volts que está sendo fornecido, I_{inst} o valor da corrente em ampères naquele instante e T_{amost} o valor do intervalo de amostragem em segundos. Após esse procedimento realiza-se o somatório dos valores E_{inst} para aquele ciclo de transmissão. Por fim, para realizar o gráfico calcula-se a média

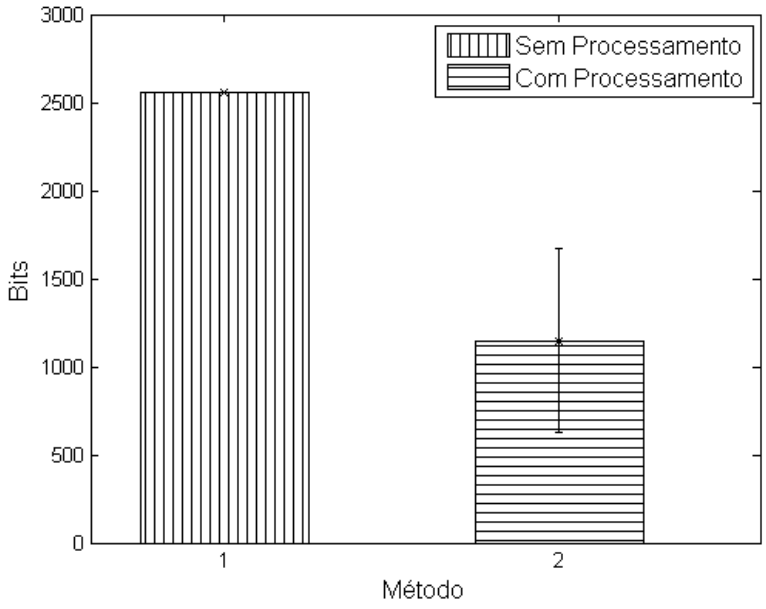

Figura 4.12: Bits Transmitidos por Ciclo de Transmissão para Aplicação de Temperatura.

para os 10 ciclos de transmissão e a barra de erro com 95% de confiança.

Ao observar as Figuras 4.10 e 4.11, percebe-se que as aplicações que utilizam a transformada conseguem reduzir significativamente o gasto de energia por ciclo de transmissão. No caso da aplicação de temperatura, ocorre uma redução em média de 50% do gasto de energia, já para o caso da aplicação de luminosidade, ocorre uma redução em média de 23%.

A maior diminuição do gasto de energia para a aplicação de temperatura já era esperada, uma vez que, conforme pode ser visto nas Figuras 4.8 e 4.9, a largura da corrente de pico para essa situação é menor do que a largura da corrente de pico para a aplicação de luminosidade.

Todavia, também ocorrerá uma maior variação desse consumo de energia, já que a barra de erro para 95% de confiança foi maior para a primeira situação. Entretanto, mesmo considerando a situação com maior gasto de energia, utilizando o valor máximo da barra de erro, para as aplicações que utilizam a transformada, ainda assim há uma redução do gasto de energia.

As Figuras 4.12 e 4.13 ilustram a quantidade de bits transmitidos por ciclo. No caso das aplicações em que não houve processamento da informação essa quantidade será constante, então a análise é focada para o caso em que ocorre o processamento.

Os gráficos das Figuras 4.12 e 4.13 confirmam o que já foi analisado anteriormente. A quantidade média de bits transmitidos para a aplicação de temperatura com DCT representa em torno de 40% dos bits transmitidos do caso em que não há o cálculo da transformada. Já para a aplicação de luminosidade, a redução dos bits a serem transmitidos não é tão significativa. Sendo assim, esta análise ratifica o que já tinha sido comentado anteriormente, ou seja, a diminuição do consumo de energia ao reduzir o montante de dados a ser transmitido.

Ao analisar a barra de erro para as duas aplicações, percebe-se que para a aplicação de tempe-

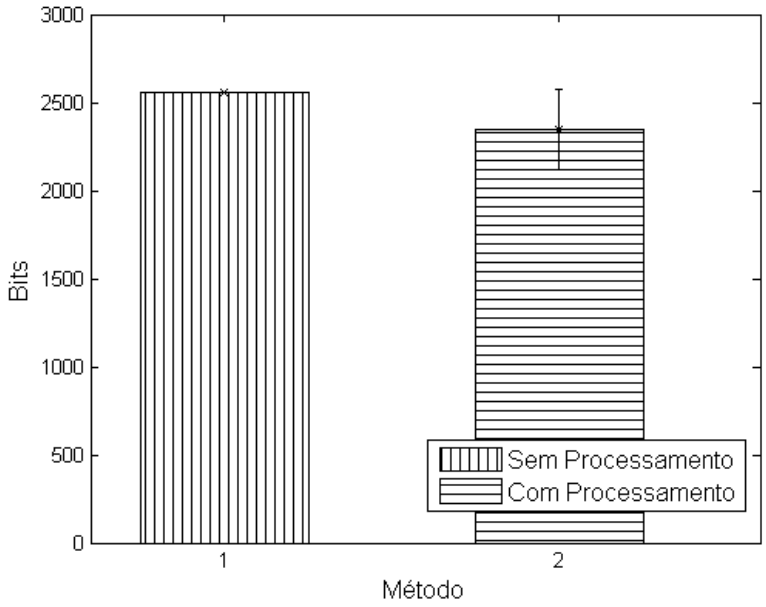

Figura 4.13: Bits Transmitidos por Ciclo de Transmissão para Aplicação de Luminosidade.

ratura a quantidade de bits transmitidos tende a ser menor ao realizar o cálculo da transformada. Entretanto, para a aplicação de luminosidade, há uma maior probabilidade de realizar a transmissão de mais bits ao realizar o cálculo da transformada. Todavia, considerando as Figuras 4.10,4.11,4.12 e 4.13, nota-se que ainda é uma boa solução realizar a transformada para diminuir o consumo de energia.

Por fim, nas Figuras 4.14 e 4.15, os sinais recebidos foram reconstruídos, para efetuar a comparação entre o sinal em que não houve processamento com o sinal em que houve processamento, a fim de verificar a diferença entre os dois. Essa análise é necessária pois possibilita descobrir se a não transmissão de coeficientes da transformada e a não transmissão da parte decimal dos coeficientes afeta significativamente a reconstrução do sinal, introduzindo erro a ponto de não reconhecê-lo. Para a reconstrução do sinal, os coeficientes que não foram transmitidos foram considerados equivalentes a zero no cálculo da transformada inversa.

Consegue-se perceber ao analisar as Figuras 4.14 e 4.15 que o sinal reconstruído quando se utiliza a transformada é praticamente idêntico ao sinal quando não há a utilização da transformada. Para ter uma comparação mais exata da diferença entre os sinais foi calculado o erro quadrático médio para as duas aplicações, obtendo os seguintes resultados:

- *Temperatura*: $3,6681.10^{-5}\ ^{\circ}C^2$
- *Luminosidade*: $6,1173.10^{-8}\ V^2$

Considerando todas as análises feitas até agora, valida-se a implementação da DCT nos *motes MICAZ*, tanto da perspectiva de consumo de energia quanto da perspectiva de reconstrução do sinal.

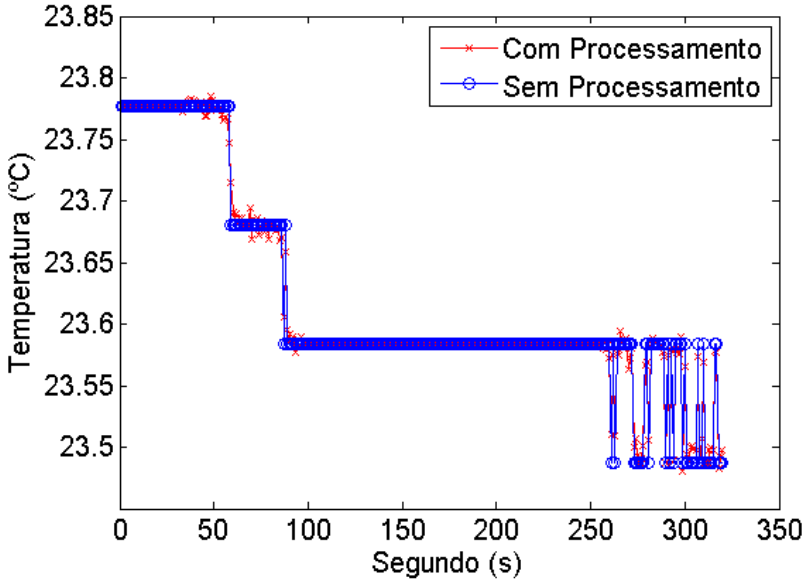

Figura 4.14: Sinal Recebido para Aplicação de Temperatura.

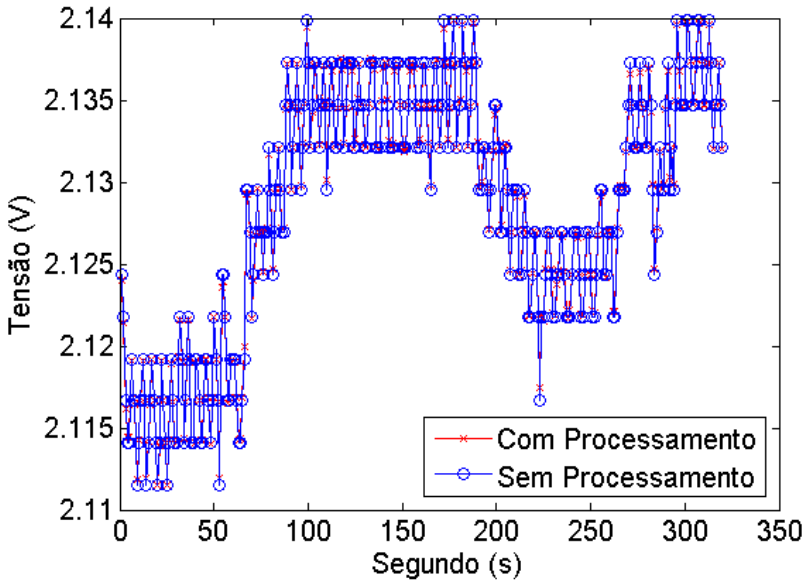

Figura 4.15: Sinal Recebido para Aplicação de Luminosidade.

4.4 Pseudo-códigos - *Matlab*

Nesta seção são apresentados os pseudo-códigos utilizados para estimar os gastos de energia para aquisição, processamento e transmissão do sinal eletrocardiográfico.

4.4.1 Sinal Eletrocardiográfico - Sem Compressão

O sinal obtido foi adquirido com uma amostragem de 360 amostras por segundo, logo o gasto vai ser calculado a cada 360 amostras, sendo então adicionado ao gasto total.

Algoritmo 1 Pseudo-código para a estimação do consumo de energia para o caso sem compressão.

entrada : $SINAL\_ECG$ adquirido através do *EcgConverter*

1: $ENERGIA\_SENS\_TOTAL \leftarrow 0$
2: $ENERGIA\_TRANS\_TOTAL \leftarrow 0$
3: $ENERGIA\_TOTAL \leftarrow 0$
4: **para** $i = 1$ até o tamanho de $SINAL\_ECG/360$ **faça**
5: $ECG\_ATUAL \leftarrow SINAL\_ECG((360 * (i-1)) + 1 : 360 * (i))$
6: $ENERGIA\_SENS \leftarrow$ calcula o gasto de aquisição de acordo com 2.4
7: $ENERGIA\_SENS\_TOTAL \leftarrow ENERGIA\_SENS + ENERGIA\_SENS\_TOTAL$
8: $BITS \leftarrow 0$
9: **para** $i = 1$ até o tamanho de ECG_ATUAL **faça**
$BITS \leftarrow BITS + quant\_bits(ECG\_ATUAL(i))$
10: **fim do para**
11: $\gamma \leftarrow$ calcula o fator de utilização de acordo com 3.2 e $BITS$
12: $ENERGIA\_TRANS \leftarrow$ calcula o gasto de transmissão de acordo com 3.1 e γ
13: $ENERGIA\_TRANS\_TOTAL \leftarrow ENERGIA\_TRANS\_TOTAL+ENERGIA\_TRANS$
14: $ENERGIA\_TOTAL \leftarrow ENERGIA\_TOTAL+ENERGIA\_SENS+ENERGIA\_TRANS$
15: **fim do para**

fim

4.4.2 Sinal Eletrocardiográfico - *Compressed Sensing*

Neste caso é calculada uma matriz binária e a partir dela é realizada a multiplicação matricial com o sinal eletrocardiográfico, sendo transmitido o resultado dessa multiplicação. A matriz binária em um caso de real implementação não precisa ser gerada no *mote*, e sim, ter seus elementos salvos na memória flash do *mote*, estando assim, pronta para uso.

Algoritmo 2 Pseudo-código para a estimação do consumo de energia para *Compressed Sensing*.

entrada : $SINAL\_ECG$ adquirido através do *EcgConverter*

1: $ENERGIA\_SENS\_TOTAL \leftarrow 0$
2: $ENERGIA\_PROC\_TOTAL \leftarrow 0$
3: $ENERGIA\_TRANS\_TOTAL \leftarrow 0$
4: $ENERGIA\_TOTAL \leftarrow 0$
5: $MATBIN \leftarrow$ gera matriz binária de acordo com teoria apresentada em 2
6: **para** $i = 1$ até o tamanho de $SINAL\_ECG/360$ **faça**

7: $ECG\_ATUAL \leftarrow SINAL\_ECG((360 * (i-1)) + 1 : 360 * (i))$
8: $ENERGIA\_SENS \leftarrow$ calcula o gasto de aquisição de acordo com 2.4
9: $ENERGIA\_SENS\_TOTAL \leftarrow ENERGIA\_SENS + ENERGIA\_SENS\_TOTAL$
10: $VET\_MULT \leftarrow MATBIN * ECG\_ATUAL$
11: $ENERGIA\_PROC \leftarrow$ calcula o gasto de processamento de acordo com 2.3 utilizando dados para filtro FIR
12: $ENERGIA\_PROC\_TOTAL \leftarrow ENERGIA\_PROC\_TOTAL + ENERGIA\_PROC$
13: $BITS \leftarrow 0$
14: **para** $i = 1$ até o tamanho de VET_MULT **faça**
$BITS \leftarrow BITS + quant\_bits(VET\_MULT(i))$
15: **fim do para**
16: $\gamma \leftarrow$ calcula o fator de utilização de acordo com 3.2 e $BITS$
17: $ENERGIA\_TRANS \leftarrow$ calcula o gasto de transmissão de acordo com 3.1 e γ
18: $ENERGIA\_TRANS\_TOTAL \leftarrow ENERGIA\_TRANS\_TOTAL+ENERGIA\_TRANS$
19: $ENERGIA\_TOTAL \leftarrow ENERGIA\_TOTAL+ENERGIA\_SENS+ENERGIA\_TRANS+ENERGIA\_PROC$
20: **fim do para**

fim

4.4.3 Sinal Eletrocardiográfico - DCT

Neste caso é calculado a DCT e ao final verifica-se quais componentes possuem maior variância, transmitindo a metade dos componentes, no caso, os que possuem maior variância. Para calcular a energia da variância deve-se ter em mente que esse cálculo é repetido para cada componente, há 360 componentes, sendo assim o cálculo do gasto deve ser multiplicado por esse fator.

Algoritmo 3 Pseudo-código para a estimação do consumo de energia para DCT.

entrada : $SINAL\_ECG$ adquirido através do *EcgConverter*
1: $ENERGIA\_SENS\_TOTAL \leftarrow 0$
2: $ENERGIA\_DCT\_TOTAL \leftarrow 0$
3: $ENERGIA\_TRANS\_TOTAL \leftarrow 0$
4: $ENERGIA\_TOTAL \leftarrow 0$
5: **para** $i = 1$ até o tamanho de $SINAL\_ECG/360$ **faça**
6: $ECG\_ATUAL \leftarrow SINAL\_ECG((360 * (i-1)) + 1 : 360 * (i))$
7: $ENERGIA\_SENS \leftarrow$ calcula o gasto de aquisição de acordo com 2.4
8: $ENERGIA\_SENS\_TOTAL \leftarrow ENERGIA\_SENS + ENERGIA\_SENS\_TOTAL$
9: $COEF = concat(COEF, dct(ECG\_ATUAL))$
10: $ENERGIA\_DCT \leftarrow$ calcula o gasto de processamento de acordo com 2.3 utilizando dados para DCT
11: $ENERGIA\_DCT\_TOTAL \leftarrow ENERGIA\_DCT\_TOTAL + ENERGIA\_DCT$
12: $ENERGIA\_TOTAL \leftarrow ENERGIA\_TOTAL+ENERGIA\_SENS+ENERGIA\_DCT$
13: **fim do para**

14: $COEF\_MAIOR \leftarrow var(COEF)$ coeficientes com maior variância
15: $ENERGIA\_VAR \leftarrow 360*$ calcula o gasto de processamento de acordo com 2.3 utilizando dados para FIR
16: $BITS \leftarrow 0$
17: **para** $i = 1$ até o tamanho de $COEF\_MAIOR$ **faça**
$BITS \leftarrow BITS + quant\_bits(COEF\_MAIOR(i))$
18: **fim do para**
19: $\gamma \leftarrow$ calcula o fator de utilização de acordo com 3.2 e $BITS$
20: $ENERGIA\_TRANS\_TOTAL \leftarrow$ calcula o gasto de transmissão de acordo com 3.1 e γ
21: $ENERGIA\_TOTAL \leftarrow ENERGIA\_TOTAL+ENERGIA\_TRANS\_TOTAL+ENERGIA\_VAR$

fim

4.5 Conclusão

Neste capítulo foram apresentados os resultados obtidos nos experimentos do *Matlab* e práticos. A partir dos experimentos do *Matlab* foi possível averiguar que o gasto de energia pela aquisição de dados possui uma relevância considerável para o gasto total do *mote*. Além disso, foi verificado que um processamento baseado somente na DCT seria a melhor solução para redução do consumo de energia do *mote* visto que essa tem um consumo de energia menor para processamento.

Baseado nos resultados dos experimentos do *Matlab*, foram realizadas implementações de aplicações que utilizassem a DCT a fim de compará-las com aplicações que não a utilizassem, realizando assim o levantamento das curvas de corrente, consumo de energia, bits transmitidos e sinais recebidos para cada situação. A partir dessas curvas foi possível verificar um melhor desempenho relativo ao consumo de energia para aplicações que utilizam a DCT, e ao comparar os sinais reconstruídos é possível verificar que os sinais das aplicações que utilizam a DCT possuem um erro quadrático médio pequeno ao comparar com o sinal em que não é utilizado a DCT.

Capítulo 5

Conclusões

Neste trabalho foi feito um estudo sobre a capacidade de processamento e consumo de energia em *motes*, visando verificar a praticidade de realizar a implementação de técnicas de compressão para reduzir o gasto de energia total nesses dispositivos.

Inicialmente, os modelos de consumo de energia foram verificados, para em seguida, mostrar as deficiências em determinados pontos, por exemplo, o gasto relacionado com a transmissão, propondo assim modificações. A partir disso, foram realizadas estimações do consumo de energia para aquisição, processamento e transmissão de sinais eletrocardiográficos a fim de verificar qual método utilizado poderia reduzir o gasto de energia total apesar de inserir um gasto com processamento. Com os resultados, foi verificado que o único processamento que resultaria na diminuição do gasto total seria um método que fosse baseado apenas na DCT.

Após essa análise, foram implementadas aplicações de temperatura e luminosidade, sendo que em um caso não ocorreria o processamento da informação e no outro ocorreria o cálculo da DCT com alguns coeficientes não sendo transmitidos. Ao comparar as duas situações, percebe-se uma economia de energia significativa para a aplicação de temperatura e a aplicação de luminosidade também obteve uma economia do gasto de energia, não tão significativo. Além do mais, foram comparados os sinais reconstruídos, podendo assim validar que apesar da não transmissão de alguns coeficientes, o sinal reconstruído da transformada possui pouco erro, quando comparado com o sinal em que não houve o cálculo da transformada.

Para os trabalhos futuros pode-se citar as seguintes modificações: armazenar os valores de cosseno dos ângulos necessários para a DCT na memória flash a fim de reduzir o processamento, calcular somente os primeiros coeficientes da DCT e verificar o impacto no consumo de energia, aumentar o limiar que define quais coeficientes não são transmitidos e verificar o impacto no gasto de energia, investigar o consumo de energia gerado pelo protocolo de transmissão, modificar o algoritmo rápido inserindo as normalizações retiradas para possibilitar o cálculo da transformada com mais coeficientes, implementar outros métodos de compressão, inserir códigos verificadores ou corretores de erro, separar os gastos provenientes da aquisição, processamento e transmissão a partir do gasto de energia total obtido no experimento prático para comparar os gastos obtidos com os modelos de consumo de energia existentes.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] MICAZ - Wireless Measurement System. `http://www.openautomation.net/uploadsproductos/micaz_datasheet.pdf`. Acessado: 15/06/2014.

[2] HEINZELMAN, W. R. et al. Energy-scalable algorithms and protocols for wireless microsensor networks. *Acoustics, Speech, and Signal Processing, 2000. ICASSP'00. Proceedings. 2000 IEEE International Conference on*, 2000.

[3] HALGAMUGE, M. N. et al. An estimation of sensor energy consumption. *Progress In Electromagnetics Research B*, v. 12, p. 259–295, 2009.

[4] WANG, Y. C.; HSIEH, Y.; TSENG, Y. C. Compression and storage schemes in a sensor network with spatial and temporal coding techniques. *Vehicular Technology Conference, 2008. VTC Spring 2008. IEEE*, 2008.

[5] HALLOUSH, R.; MISRA, K.; RADHA, H. Practical distributed video coding over visual sensors. *Picture Coding Symposium, 2009. PCS 2009*, 2009.

[6] LEVIS, P.; GAY, D. *TinyOS Programming*. [S.l.]: Cambridge University Press, 2009.

[7] MALVAR, H. S. *Signal Processing with Lapped Transforms*. [S.l.]: Artech House, Inc., 1992.

[8] ANSARI-RAM, F.; HOSSEINI-KHAYAT, S. Ecg signal compression using compressed sensing with nonuniform binary matrices. *Artificial Intelligence and Signal Processing (AISP), 2012 16th CSI International Symposium on. IEEE*, 2012.

[9] AHMED, N.; MILNE, P. J.; HARRIS, S. G. Electrocardiographic data compression via orthogonal transforms. *Biomedical Engineering, IEEE Transactions on*, 1975.

[10] MIT-BIH Arrhythmia Database. `http://www.physionet.org/physiobank/database/mitdb/`. Acessado: 16/06/2014.

[11] PS25255 - EPIC Ultra High Impedance ECG Sensor. `http://webshop.atlantikelektronik.de/Webpage/PS25255.pdf`. Acessado: 15/06/2014.

[12] ATMEL. *8-bit Atmel Microcontroller with 128KBytes In-System Programmable Flash*. [S.l.]: Atmel. `http://www.atmel.com/Images/doc2467.pdf`.

[13] MOTE Processor Radio & Mote Interface Board User Manual. https://www.cs.wmich.edu/gupta/teaching/cs5950/fall2011/mica%20pinouts%20from%20memsic%20mpr-mib_series_users_manual%207430-0021-09_a-t.pdf. Acessado: 15/06/2014.

[14] INSTRUMENTS, T. *2.4 GHz IEEE 802.15.4/ZigBee-ready RF Transceiver*. [S.l.]: Texas Instruments. http://www.ti.com/lit/ds/symlink/cc2420.pdf.

[15] MTS/MDA Sensor Board Users Manual. http://www.investigacion.frc.utn.edu.ar/sensores/Equipamiento/Wireless/MTS-MDA_Series_Users_Manual.pdf. Acessado: 15/06/2014.

[16] AGILENT. *Agilent 34410A/11A 6 1/2 Digit Multimeter(includes the L4411A 1U DMM)*. [S.l.]: Agilent. http://cp.literature.agilent.com/litweb/pdf/34410-90001.pdf.

[17] MULTíMETRO Digital Hikari HM-1000. http://www.hikariferramentas.com.br/produtos/?ip=86&Mult%EDmetro%20Digital%20Hikari%20HM-1000. Acessado: 16/06/2014.

[18] YSI Precision Thermistors & Probes. http://www.advindsys.com/Manuals/YSIManuals/YSICatalog-1998.pdf. Acessado: 16/06/2014.

[19] MEMSIC. *MIB 520 - USB Interface Board*. [S.l.]: Memsic. http://www.memsic.com/userfiles/files/Datasheets/WSN/6020-0091-04_a_mib520cb-t.pdf.

[20] RAZZAQUE, M. A.; DOBSON, S. Energy-efficient sensing in wireless sensor networks using compressed sensing. *Sensors, v. 14, n. 2*, 2014.

ANEXOS

I. CÓDIGOS IMPLEMENTADOS

Todos os códigos desenvolvidos neste trabalho estão disponíveis em `https://github.com/savioneves1/final`.